What Is Mind Design?

1

John Haugeland
1996

MIND DESIGN is the endeavor to understand *mind* (thinking, intellect) in terms of its *design* (how it is built, how it works). It amounts, therefore, to a kind of cognitive psychology. But it is oriented more toward structure and mechanism than toward correlation or law, more toward the "how" than the "what", than is traditional empirical psychology. An "experiment" in mind design is more often an effort to *build* something and make it work, than to observe or analyze what already exists. Thus, the field of artificial intelligence (AI), the attempt to construct intelligent artifacts, systems with minds of their own, lies at the heart of mind design. Of course, natural intelligence, especially human intelligence, remains the final object of investigation, the phenomenon eventually to be understood. What is distinctive is not the goal but rather the means to it. Mind design is *psychology by reverse engineering*.

Though the idea of intelligent artifacts is as old as Greek mythology, and a familiar staple of fantasy fiction, it has been taken seriously as science for scarcely two generations. And the reason is not far to seek: pending several conceptual and technical breakthroughs, no one had a clue how to proceed. Even as the pioneers were striking boldly into the unknown, much of what they were really up to remained unclear, both to themselves and to others; and some still does. Accordingly, mind design has always been an area of *philosophical* interest, an area in which the conceptual foundations—the very questions to ask, and what would count as an answer—have remained unusually fluid and controversial.

The essays collected here span the history of the field since its inception (though with emphasis on more recent developments). The authors are about evenly divided between philosophers and scientists. Yet, all of the essays are "philosophical", in that they address fundamental issues and basic concepts; at the same time, nearly all are also "scientific" in that they are technically sophisticated and concerned with the achievements and challenges of concrete empirical research.

Several major trends and schools of thought are represented, often explicitly disputing with one another. In their juxtaposition, therefore, not only the lay of the land, its principal peaks and valleys, but also its current movement, its still active fault lines, can come into view.

By way of introduction, I shall try in what follows to articulate a handful of the fundamental ideas that have made all this possible.

1 Perspectives and things

None of the present authors believes that intelligence depends on anything immaterial or supernatural, such as a vital spirit or an immortal soul. Thus, they are all *materialists* in at least the minimal sense of supposing that matter, suitably selected and arranged, suffices for intelligence. The question is: How?

It can seem incredible to suggest that mind is "nothing but" matter in motion. Are we to imagine all those little atoms thinking deep thoughts as they careen past one another in the thermal chaos? Or, if not one by one, then maybe collectively, by the zillions? The answer to this puzzle is to realize that things can be viewed from different *perspectives* (or described in different terms)—and, when we look differently, what we are able to see is also different. For instance, what is a coarse weave of frayed strands when viewed under a microscope is a shiny silk scarf seen in a store window. What is a marvellous old clockwork in the eyes of an antique restorer is a few cents' worth of brass, seen as scrap metal. Likewise, so the idea goes, what is mere atoms in the void from one point of view can be an intelligent system from another.

Of course, you can't look at anything in just any way you please—at least, not and be right about it. A scrap dealer couldn't see a wooden stool as a few cents' worth of brass, since it isn't brass; the antiquarian couldn't see a brass monkey as a clockwork, since it doesn't work like a clock. Awkwardly, however, these two points taken together seem to create a dilemma. According to the first, what something is—coarse or fine, clockwork or scrap metal—depends on how you look at it. But, according to the second, how you can *rightly* look at something (or describe it) depends on what it is. Which comes first, one wants to ask, seeing or being?

Clearly, there's something wrong with that question. What something is and how it can rightly be regarded are not essentially distinct; neither comes before the other, because they are the same. The advantage of emphasizing perspective, nevertheless, is that it highlights the following question: What *constrains* how something can rightly be regarded or described (and thus determines what it is)? This is important, because the answer will be different for different kinds of perspective or description—as our examples already illustrate. Sometimes, what something is is determined by its shape or form (at the relevant level of detail); sometimes it is determined by what it's made of; and sometimes by how it works or even just what it does. Which—if any—of these could determine whether something is (rightly regarded or described as) *intelligent*?

1.1 The Turing test

In 1950, the pioneering computer scientist A. M. Turing suggested that intelligence is a matter of behavior or behavioral capacity: whether a system has a mind, or how intelligent it is, is determined by what it can and cannot do. Most materialist philosophers and cognitive scientists now accept this general idea (though John Searle is an exception). Turing also proposed a pragmatic criterion or test of what a system can do that would be sufficient to show that it is intelligent. (He did not claim that a system would not be intelligent if it could not pass his test; only that it would be if it could.) This test, now called the *Turing test*, is controversial in various ways, but remains widely respected in spirit.

Turing cast his test in terms of simulation or imitation: a nonhuman system will be deemed intelligent if it acts so like an ordinary person *in certain respects* that other ordinary people can't tell (from these actions alone) that it isn't one. But the imitation idea itself isn't the important part of Turing's proposal. What's important is rather the specific sort of behavior that Turing chose for his test: he specified *verbal* behavior. A system is surely intelligent, he said, if it can carry on an ordinary conversation like an ordinary person (via electronic means, to avoid any influence due to appearance, tone of voice, and so on).

This is a daring and radical simplification. There are many ways in which intelligence is manifested. Why single out *talking* for special emphasis? Remember: Turing didn't suggest that talking in this way is required to demonstrate intelligence, only that it's sufficient. So there's no worry about the test being too hard; the only question is whether it might be too lenient. We know, for instance, that there are systems that can regulate temperatures, generate intricate rhythms, or even fly airplanes without being, in any serious sense, intelligent. Why couldn't the ability to carry on ordinary conversations be like that?

Turing's answer is elegant and deep: talking is unique among intelligent abilities because it gathers within itself, at one remove, all others. One cannot generate rhythms or fly airplanes "about" talking, but one certainly can *talk about* rhythms and flying—not to mention poetry, sports, science, cooking, love, politics, and so on—*and*, if one doesn't know what one is talking about, it will soon become painfully obvious. Talking is not merely one intelligent ability among others, but also, and essentially, the ability to *express* intelligently a great many (maybe all) other intelligent abilities. And, without *having* those abilities in fact, at least to some degree, one cannot talk intelligently about them. That's why Turing's test is so compelling and powerful.

On the other hand, even if not too easy, there is nevertheless a sense in which the test does obscure certain real difficulties. By concentrating on conversational ability, which can be exhibited entirely in writing (say, via computer terminals), the Turing test completely ignores any issues of real-world perception and action. Yet these turn out to be extraordinarily difficult to achieve artificially at any plausible level of sophistication. And, what may be worse, ignoring real-time environmental interaction distorts a system designer's assumptions about how intelligent systems are related to the world more generally. For instance, if a system has to deal or cope with things around it, but is not continually tracking them externally, then it will need somehow to "keep track of" or *represent* them internally. Thus, neglect of perception and action can lead to an overemphasis on representation and internal modeling.

1.2 Intentionality

"Intentionality", said Franz Brentano (1874/1973), "is the mark of the mental." By this he meant that everything mental has intentionality, and nothing else does (except in a derivative or second-hand way), and, finally, that this fact is the *definition of the mental*. 'Intentional' is used here in a medieval sense that harks back to the original Latin meaning of "stretching toward" something; it is not limited to things like plans and purposes, but applies to all kinds of mental acts. More specifically, intentionality is the character of one thing being "of" or "about" something else, for instance by representing it, describing it, referring to it, aiming at it, and so on. Thus, intending in the narrower modern sense (planning) is also intentional in Brentano's broader and older sense, but much else is as well, such as believing, wanting, remembering, imagining, fearing, and the like.

Intentionality is peculiar and perplexing. It looks on the face of it to be a relation between two things. My belief that Cairo is hot is intentional because it is *about* Cairo (and/or its being hot). That which an intentional act or state is about (Cairo or its being hot, say) is called its *intentional object*. (It is this intentional object that the intentional state "stretches toward".) Likewise, my desire for a certain shirt, my imagining a party on a certain date, my fear of dogs in general, would be "about"—that is, have as their intentional objects—that shirt, a party on that date, and dogs in general. Indeed, *having* an object in this way is another way of explaining intentionality; and such "having" seems to be a relation, namely between the state and its object.

But, if it's a relation, it's a relation like no other. Being-inside-of is a typical relation. Now notice this: if it is a fact about one thing that it is inside of another, then not only that first thing, but also the second has to *exist*; *X* cannot be inside of *Y*, or indeed be related to *Y* in any other way, if *Y* does not exist. This is true of relations quite generally; but it is *not* true of intentionality. I can perfectly well imagine a party on a certain date, and also have beliefs, desires, and fears about it, even though there is (was, will be) no such party. Of course, those beliefs would be false, and those hopes and fears unfulfilled; but they would be intentional—be about, or "have", those objects—all the same.

It is this puzzling ability to have something as an object, whether or not that something actually exists, that caught Brentano's attention. Brentano was no materialist: he thought that mental phenomena were one kind of entity, and material or physical phenomena were a completely different kind. And he could not see how *any* merely material or physical thing could be *in fact* related to another, if the latter didn't exist; yet *every* mental state (belief, desire, and so on) has this possibility. So intentionality is the definitive mark of the mental.

Daniel C. Dennett accepts Brentano's definition of the mental, but proposes a materialist way to view intentionality. Dennett, like Turing, thinks intelligence is a matter of how a system behaves; but, unlike Turing, he also has a worked-out account of what it is about (some) behavior that makes it intelligent—or, in Brentano's terms, makes it the behavior of a system with intentional (that is, *mental*) states. The idea has two parts: (i) behavior should be understood not in isolation but in *context* and as part of a consistent *pattern* of behavior (this is often called "holism"); and (ii) for some systems, a consistent pattern of behavior in context can be construed as *rational* (such construing is often called "interpretation").[1]

Rationality here means: acting so as best to satisfy your goals overall, given what you know and can tell about your situation. Subject to this constraint, we can surmise what a system wants and believes by watching what it does—but, of course, not in isolation. From all you can tell in isolation, a single bit of behavior might be manifesting any number of different beliefs and/or desires, or none at all. Only when you see a *consistent pattern of rational behavior*, manifesting the *same* cognitive states and capacities repeatedly, in various combinations, are you justified in saying that *those* are the states and capacities that this system has—or even that it has *any* cognitive states or capacities at all. "Rationality", Dennett says (1971/78, p. 19), "is the mother of intention."

This is a prime example of the above point about *perspective*. The constraint on whether something can rightly be regarded as having intentional states is, according to Dennett, not its shape or what it is made of, but rather what it does—more specifically, a consistently rational pattern in what it does. We infer that a rabbit can tell a fox from another rabbit, always wanting to get away from the one but not the other, from having observed it behave accordingly time and again, under various conditions. Thus, on a given occasion, we impute to the rabbit *intentional* states (beliefs and desires) *about* a particular fox, on the basis not only of its current behavior but also of the pattern in its behavior over time. The consistent pattern lends both specificity and credibility to the respective individual attributions.

Dennett calls this perspective the *intentional stance* and the entities so regarded *intentional systems*. If the stance is to have any conviction in any particular case, the pattern on which it depends had better be broad and reliable; but it needn't be perfect. Compare a crystal: the pattern in the atomic lattice had better be broad and reliable, if the sample is to be a crystal at all; but it needn't be perfect. Indeed, the very idea of a *flaw* in a crystal is made intelligible by the regularity of the pattern around it; only insofar as *most* of the lattice is regular, can particular parts be deemed flawed in determinate ways. Likewise for the intentional stance: only because the rabbit behaves rationally almost always, could we ever say on a particular occasion that it happened to be *wrong*—had *mistaken* another rabbit (or a bush, or a shadow) for a fox, say. False beliefs and unfulfilled hopes are intelligible as isolated lapses in an overall consistent pattern, like flaws in a crystal. This is how a specific intentional state can rightly be attributed, even though its supposed intentional object doesn't exist—and thus is Dennett's answer to Brentano's puzzle.

1.3 Original intentionality

Many material things that aren't intentional systems are nevertheless "about" other things—including, sometimes, things that don't exist. Written sentences and stories, for instance, are in some sense material; yet they are often about fictional characters and events. Even pictures and maps can represent nonexistent scenes and places. Of course, Brentano knew this, and so does Dennett. But they can say that this sort of intentionality is only *derivative*. Here's the idea: sentence inscriptions—ink marks on a page, say—are only "about" anything because we (or other intelligent users) *mean* them that way. Their intentionality is second-hand, borrowed or derived from the intentionality that those users already have.

So, a sentence like "Santa lives at the North Pole", or a picture of him or a map of his travels, can be "about" Santa (who, alas, doesn't exist), but *only because* we can *think* that he lives there, and *imagine* what he looks like and where he goes. It's really *our* intentionality that these artifacts have, second-hand, because we use them to *express* it. Our intentionality itself, on the other hand, cannot be likewise derivative: it must be *original*. ('Original', here, just means *not* derivative, not borrowed from somewhere else. If there is any intentionality at all, at least some of it must be original; it can't all be derivative.)

The problem for mind design is that artificial intelligence systems, like sentences and pictures, are also artifacts. So it can seem that their intentionality too must always be derivative—borrowed from their designers or users, presumably—and never original. Yet, if the project of designing and building a system with a mind of its own is ever really to succeed, then it must be possible for an artificial system to have genuine *original* intentionality, just as we do. Is that possible?

Think again about people and sentences, with their original and derivative intentionality, respectively. What's the reason for that difference? Is it really that sentences are artifacts, whereas people are not, or might it be something else? Here's another candidate. Sentences don't *do* anything with what they mean: they never pursue goals, draw conclusions, make plans, answer questions, let alone *care* whether they are right or wrong about the world—they just sit there, utterly inert and heedless. A person, by contrast, relies on what he or she believes and wants in order to make sensible choices and act efficiently; and this entails, in turn, an ongoing concern about whether those beliefs are really true, those goals really beneficial, and so on. In other words, real beliefs and desires are integrally involved in a rational, active existence,

intelligently engaged with its environment. Maybe this active, rational engagement is more pertinent to whether the intentionality is original or not than is any question of natural or artificial origin.

Clearly, this is what Dennett's approach implies. An intentional system, by his lights, is just one that exhibits an appropriate pattern of consistently rational *behavior*—that is, active engagement with the world. If an artificial system can be produced that behaves on its own in a rational manner, consistently enough and in a suitable variety of circumstances (remember, it doesn't have to be flawless), then it has *original* intentionality—it has a mind of its own, just as we do.

On the other hand, Dennett's account is completely silent about how, or even whether, such a system could actually be designed and built. Intentionality, according to Dennett, depends entirely and exclusively on a certain sort of pattern in a system's behavior; internal structure and mechanism (if any) are quite beside the point. For scientific mind design, however, the question of how it actually works (and so, how it could be built) is absolutely central—and that brings us to computers.

2 Computers

Computers are important to scientific mind design in two fundamentally different ways. The first is what inspired Turing long ago, and a number of other scientists much more recently. But the second is what really launched AI and gave it its first serious hope of success. In order to understand these respective roles, and how they differ, it will first be necessary to grasp the notion of 'computer' at an essential level.

2.1 Formal systems

A formal system is like a game in which tokens are manipulated according to definite rules, in order to see what configurations can be obtained. In fact, many familiar games—among them chess, checkers, tic-tac-toe, and go—simply *are* formal systems. But there are also many games that are not formal systems, and many formal systems that are not games. Among the former are games like marbles, tiddlywinks, billiards, and baseball; and among the latter are a number of systems studied by logicians, computer scientists, and linguists.

This is not the place to attempt a full definition of formal systems; but three essential features can capture the basic idea: (i) they are (as indicated above) token-manipulation systems; (ii) they are digital; and

(iii) they are medium independent. It will be worth a moment to spell out what each of these means.

Token-manipulation systems. To say that a formal system is a token-manipulation system is to say that you can define it *completely* by specifying three things:

(1) a set of types of formal tokens or pieces;

(2) one or more allowable starting positions—that is, initial formal arrangements of tokens of these types; and

(3) a set of formal rules specifying how such formal arrangements may or must be changed into others.

This definition is meant to imply that token-manipulation systems are entirely *self-contained.* In particular, the formality of the rules is twofold: (i) they specify *only* the allowable next formal arrangements of tokens, and (ii) they specify these in terms *only* of the current formal arrangement—nothing else is *formally* relevant at all.

So take chess, for example. There are twelve types of piece, six of each color. There is only one allowable starting position, namely one in which thirty-two pieces of those twelve types are placed in a certain way on an eight-by-eight array of squares. The rules specifying how the positions change are simply the rules specifying how the pieces move, disappear (get captured), or change type (get promoted). (In chess, new pieces are never added to the position; but that's a further kind of move in other formal games—such as go.) Finally, notice that chess is entirely self-contained: nothing is ever relevant to what moves would be legal other than the current chess position itself.[2]

And every student of formal logic is familiar with at least one logical system as a token-manipulation game. Here's one obvious way it can go (there are many others): the kinds of logical symbol are the types, and the marks that you actually make on paper are the tokens of those types; the allowable starting positions are sets of well-formed formulae (taken as premises); and the formal rules are the inference rules specifying steps—that is, further formulae that you write down and add to the current position—in formally valid inferences. The fact that this is called *formal* logic is, of course, no accident.

Digital systems. Digitalness is a characteristic of certain techniques (methods, devices) for *making* things, and then (later) *identifying* what was made. A familiar example of such a technique is writing something down and later reading it. The thing written or made is supposed to be

of a specified type (from some set of possible types), and identifying it later is telling what type that was. So maybe you're supposed to write down specified letters of the alphabet; and then my job is to tell, on the basis of what you produce, which letters you were supposed to write. Then the question is: how well can I do that? How good are the later identifications at recovering the prior specifications?

Such a technique is *digital* if it is positive and reliable. It is *positive* if the reidentification can be *absolutely perfect*. A positive technique is *reliable* if it not only can be perfect, but almost always is. This bears some thought. We're accustomed to the idea that nothing—at least, nothing mundane and real-worldly—is ever quite *perfect*. Perfection is an ideal, never fully attainable in practice. Yet the definition of 'digital' requires that perfection be not only possible, but reliably achievable.

Everything turns on what counts as success. Compare two tasks, each involving a penny and an eight-inch checkerboard. The first asks you to place the penny *exactly* 0.43747 inches in from the nearest edge of the board, and 0.18761 inches from the left; the second asks you to put it *somewhere* in the fourth rank (row) and the second file (column from the left). Of course, achieving the first would also achieve the second. But the first task is strictly impossible—that is, it can never actually be achieved, but at best approximated. The second task, on the other hand, can in fact be carried out *absolutely perfectly*—it's not even hard. And the reason is easy to see: any number of slightly different actual positions would equally well count as *complete* success—because the penny only has to be *somewhere* within the specified square.

Chess is digital: if one player produces a chess position (or move), then the other player can reliably identify it *perfectly*. Chess positions and moves are like the second task with the penny: slight differences in the physical locations of the figurines aren't differences at all from the chess point of view—that is, in the positions of the chess pieces. Checkers, go, and tic-tac-toe are like chess in this way, but baseball and billiards are not. In the latter, unlike the former, arbitrarily small differences in the exact position, velocity, smoothness, elasticity, or whatever, of some physical object can make a significant difference to the game. Digital systems, though concrete and material, are insulated from such physical vicissitudes.

MEDIUM INDEPENDENCE. A concrete system is medium independent if what it is does not depend on what physical "medium" it is made of or implemented in. Of course, it has to be implemented in *something*;

and, moreover, that something has to support whatever structure or form is necessary for the kind of system in question. But, apart from this generic prerequisite, nothing specific about the medium matters (except, perhaps, for extraneous reasons of convenience). In this sense, only the *form* of a formal system is significant, not its matter.

Chess, for instance, is medium independent. Chess pieces can be made of wood, plastic, ivory, onyx, or whatever you want, just as long as they are sufficiently stable (they don't melt or crawl around) and are movable by the players. You can play chess with patterns of light on a video screen, with symbols drawn in the sand, or even—if you're rich and eccentric enough—with fleets of helicopters operated by radio control. But you can't play chess with live frogs (they won't sit still), shapes traced in the water (they won't last), or mountain tops (nobody can move them). Essentially similar points can be made about logical symbolism and all other formal systems.

By contrast, what you can light a fire, feed a family, or wire a circuit with is not medium independent, because whether something is flammable, edible, or electrically conductive depends not just on its form but also on what it's made of. Nor are billiards or baseball independent of their media: what the balls (and bats and playing surfaces) are made of is quite important and carefully regulated. Billiard balls can indeed be made either of ivory or of (certain special) plastics, but hardly of wood or onyx. And you couldn't play billiards or baseball with helicopters or shapes in the sand to save your life. The reason is that, unlike chess and other formal systems, in these games the details of the physical interactions of the balls and other equipment make an important difference: how they bounce, how much friction there is, how much energy it takes to make them go a certain distance, and so on.

2.2 Automatic formal systems

An *automatic* formal system is a formal system that "moves" by itself. More precisely, it is a physical device or machine such that:

(1) some configurations of its parts or states can be regarded as the tokens and positions of some formal system; and

(2) in its normal operation, it automatically manipulates these tokens in accord with the rules of that system.

So it's like a set of chess pieces that hop around the board, abiding by the rules, all by themselves, or like a magical pencil that writes out formally correct logical derivations, without the guidance of any logician.

Of course, this is exactly what computers are, seen from a formal perspective. But, if we are to appreciate properly their importance for mind design, several fundamental facts and features will need further elaboration—among them the notions of implementation and universality, algorithmic and heuristic procedures, and digital simulation.

Implementation and universality. Perhaps the most basic idea of computer science is that you can use one automatic formal system to *implement* another. This is what *programming* is. Instead of building some special computer out of hardware, you build it out of software; that is, you write a program for a "general purpose" computer (which you already have) that will make it act exactly as if it were the special computer that you need. One computer so implements another when:

(1) some configurations of tokens and positions of the former can be regarded as the tokens and positions of the latter; and

(2) as the former follows its own rules, it automatically manipulates those tokens of the latter in accord with the latter's rules.

In general, those configurations that are being regarded as tokens and positions of the special computer are themselves only a fraction of the tokens and positions of the general computer. The remainder (which may be the majority) are the program. The general computer follows its own rules with regard to *all* of its tokens; but the program tokens are so arranged that the net effect is to manipulate the configurations implementing the tokens of the special computer in exactly the way required by its rules.

This is complicated to describe, never mind actually to achieve; and the question arises how often such implementation is possible in principle. The answer is as surprising as it is consequential. In 1937, A. M. Turing—the same Turing we met earlier in our discussion of intelligence—showed, in effect, that it is *always* possible. Put somewhat more carefully, he showed that there are some computing machines—which he called *universal* machines—that can implement *any* well-defined automatic formal system whatsoever, provided only that they have enough storage capacity and time. Not only that, he showed also that universal machines can be amazingly simple; and he gave a complete design specification for one.

Every ordinary (programmable) computer is a universal machine in Turing's sense. In other words, the computer on your desk, given the right program and enough memory, could be made equivalent to any

computer that is possible at all, in every respect except speed. Anything any computer can do, yours can too, in principle. Indeed, the machine on your desk can be (and usually is) lots of computers at once. From one point of view, it is a "hardware" computer modifying, according to strict formal rules, complex patterns of tiny voltage tokens often called "bits". Viewed another way, it is simultaneously a completely different system that shuffles machine-language words called "op-codes", "data" and "addresses". And, depending on what you're up to, it may also *be* a word processor, a spell checker, a macro interpreter, and/or whatever.

Algorithms and heuristics. Often a specific computer is designed and built (or programed) for a particular purpose: there will be some complicated rearrangement of tokens that it would be valuable to bring about automatically. Typically, a designer works with facilities that can carry out simple rearrangements easily, and the job is to find a combination of them (usually a sequence of steps) that will collectively achieve the desired result. Now there are two basic kinds of case, depending mainly on the character of the assigned task.

In many cases, the designer is able to implement a procedure that is guaranteed always to work—that is, to effect the desired rearrangement, regardless of the input, in a finite amount of time. Suppose, for instance, that the input is always a list of English words, and the desired rearrangement is to put them in alphabetical order. There are known procedures that are guaranteed to alphabetize any given list in finite time. Such procedures, ones that are sure to succeed in finite time, are called *algorithms*. Many important computational problems can be solved algorithmically.

But many others cannot, for theoretical or practical reasons. The task, for instance, might be to find the optimal move in any given chess position. Technically, chess is finite; so, theoretically, it would be possible to check every possible outcome of every possible move, and thus choose flawlessly, on the basis of complete information. But, in fact, even if the entire planet Earth were one huge computer built with the best current technology, it could not solve this problem even once in the life of the Solar System. So chess by brute force is impractical. But that, obviously, does not mean that machines can't come up with good chess moves. How do they do that?

They rely on general estimates and rules of thumb: procedures that, while not guaranteed to give the right answer every time, are fairly reliable most of the time. Such procedures are called *heuristics*. In the

case of chess, sensible heuristics involve looking ahead a few moves in various directions and then evaluating factors like number and kind of pieces, mobility, control of the center, pawn coordination, and so on. These are not infallible measures of the strength of chess positions; but, in combination, they can be pretty good. This is how chess-playing computers work—and likewise many other machines that deal with problems for which there are no known algorithmic solutions.

The possibility of heuristic procedures on computers is sometimes confusing. In one sense, every digital computation (that does not consult a randomizer) is algorithmic; so how can any of them be heuristic? The answer is again a matter of perspective. Whether any given procedure is algorithmic or heuristic depends on how you describe the task. One and the same procedure can be an algorithm, when described as counting up the number and kinds of pieces, but a mere heuristic rule of thumb, when described as estimating the strength of a position.

This is the resolution of another common confusion as well. It is often said that computers never make mistakes (unless there is a bug in some program or a hardware malfunction). Yet anybody who has ever played chess against a small chess computer knows that it makes plenty of mistakes. But this is just that same issue about how you describe the task. Even that cheap toy is executing the algorithms that implement its heuristics flawlessly every time; seen that way, it never makes a mistake. It's just that those heuristics aren't very sophisticated; so, seen as a chess player, the same system makes lots of mistakes.

Digital simulation. One important practical application of computers isn't really token manipulation at all, except as a means to an end. You see this in your own computer all the time. Word processors and spreadsheets literally work with digital tokens: letters and numerals. But image processors do not: pictures are *not* digital. Rather, as everybody knows, they are "digitized". That is, they are divided up into fine enough dots and gradations that the increments are barely perceptible, and the result looks smooth and continuous. Nevertheless, the computer can store and modify them because—*redescribed*—those pixels are all just digital numerals.

The same thing can be done with dynamic systems: systems whose states interact and change in regular ways over time. If the relevant variables and relationships are known, then time can be divided into small intervals too, and the progress of the system computed, step by tiny step. This is called *digital simulation*. The most famous real-world

example of it is the massive effort to predict the weather by simulating the Earth's atmosphere. But engineers and scientists—including, as we shall see, many cognitive scientists—rely on digital simulation of nondigital systems all the time.

2.3 Computers and intelligence

Turing (1950 [chapter 2 in this volume], 442 [38]) predicted—falsely, as we now know, but not foolishly—that by the year 2000 there would be computers that could pass his test for intelligence. This was before any serious work, theoretical or practical, had begun on artificial intelligence at all. On what, then, did he base his prediction? He doesn't really say (apart from an estimate—quite low—of how much storage computers would then have). But I think we can see what moved him.

In Turing's test, the only relevant inputs and outputs are *words*—all of which are (among other things) formal tokens. So the capacity of human beings that is to be matched is effectively a formal input/output function. But Turing himself had shown, thirteen years earlier, that *any* formal input/output function from a certain very broad category could be implemented in a routine universal machine, provided only that it had enough memory and time (or speed)—and those, he thought, would be available by century's end.

Now, this isn't really a proof, even setting aside the assumptions about size and speed, because Turing did not (and could not) show that the human verbal input/output function fell into that broad category of functions to which his theorem applied. But he had excellent reason to believe that any function computable by any *digital* mechanism would fall into that category; and he was convinced that there is nothing immaterial or supernatural in human beings. The only alternative remaining would seem to be *non*digital mechanisms; and those he believed could be digitally simulated.

Notice that there is *nothing* in this argument about how the mind might actually work—nothing about actual *mind design*. There's just an assumption that there must be *some* (nonmagical) way that it works, and that, whatever that way is, a computer can either implement it or simulate it. In the subsequent history of artificial intelligence, on the other hand, a number of very concrete proposals have been made about the actual design of human (and/or other) minds. Almost all of these fall into one or the other of two broad groups: those that take seriously the idea that the mind itself is essentially a digital computer (of a particular sort), and those that reject that idea.

3 GOFAI

The first approach is what I call "good old-fashioned AI", or *GOFAI*. (It is also sometimes called "classical" or "symbol-manipulation" or even "language-of-thought" AI.) Research in the GOFAI tradition dominated the field from the mid-fifties through at least the mid-eighties, and for a very good reason: it was (and still is) a well-articulated view of the mechanisms of intelligence that is both intuitively plausible and eminently realizable. According to this view, the mind just *is* a computer with certain special characteristics—namely, one with internal states and processes that can be regarded as explicit *thinking* or *reasoning*. In order to understand the immense plausibility and power of this GOFAI idea, we will need to see how a computer could properly be regarded in this way.

3.1 Interpreted formal systems

The idea of a formal system emerged first in mathematics, and was inspired by arithmetic and algebra. When people solve arithmetic or algebraic problems, they manipulate tokens according to definite rules, sort of like a game. But there is a profound difference between these tokens and, say, the pieces on a chess board: they *mean* something. Numerals, for instance, represent numbers (either of specified items or in the abstract), while arithmetic signs represent operations on or relationships among those numbers. (Tokens that mean something in this way are often called *symbols*.) Chess pieces, checkers, and go stones, by contrast, represent nothing: they are not symbols at all, but *merely* formal game tokens.

The rules according to which the tokens in a mathematical system may be manipulated and what those tokens mean are closely related. A simple example will bring this out. Suppose someone is playing a formal game with the first fifteen letters of the alphabet. The rules of this game are very restrictive: every starting position consists of a string of letters ending in 'A' (though not every such string is legal); and, for each starting position, there is one and only one legal move—which is to append a *particular* string of letters after the 'A' (and then the game is over). The question is: What (if anything) is going on here?

Suppose it occurs to you that the letters might be just an oddball notation for the familiar digits and signs of ordinary arithmetic. There are, however, over a trillion possible ways to translate fifteen letters into fifteen digits and signs. How could you decide which—*if any*—is

Eight sample games (before translation):

Starting position	*Legal move*	*Starting position*	*Legal move*
OEO A	N	MMCN A	JJ
NIBM A	G	OODF A	OO
HCHCH A	KON	IDL A	M
KEKDOF A	F	NBN A	O

First translation scheme:			Sample games, by first translation:			
A ⇒ 1	F ⇒ 6	K ⇒ +	=5=1	÷	××3÷1	00
B ⇒ 2	G ⇒ 7	L ⇒ −	÷92×1	7	==46 1	==
C ⇒ 3	H ⇒ 8	M ⇒ ×	83838 1	+=÷	94−1	×
D ⇒ 4	I ⇒ 9	N ⇒ ÷	+5+4=6 1	6	÷2÷1	=
E ⇒ 5	J ⇒ 0	O ⇒ =				

Second translation scheme:			Sample games, by second translation:			
A ⇒ =	F ⇒ 0	K ⇒ 5	9÷9=	8	77−8=	44
B ⇒ +	G ⇒ 1	L ⇒ 6	83+7=	1	99×0=	99
C ⇒ −	H ⇒ 2	M ⇒ 7	2−2−2=	598	3×6=	2
D ⇒ ×	I ⇒ 3	N ⇒ 8	5÷5×90=	0	8+8=	9
E ⇒ ÷	J ⇒ 4	O ⇒ 9				

Third translation scheme:			Sample games, by third translation:			
A ⇒ =	F ⇒ 0	K ⇒ 5	1+1=	2	33×2=	66
B ⇒ ÷	G ⇒ 9	L ⇒ 4	27÷3=	9	11−0=	11
C ⇒ ×	H ⇒ 8	M ⇒ 3	8×8×8=	512	7−4=	3
D ⇒ −	I ⇒ 7	N ⇒ 2	5+5−10=	0	2÷2=	1
E ⇒ +	J ⇒ 6	O ⇒ 1				

Table 1.1: Letter game and three different translation schemes.

the "right" way? The problem is illustrated in table 1.1. The first row gives eight sample games, each legal according to the rules. The next three rows each give a possible translation scheme, and show how the eight samples would come out according to that scheme.

The differences are conspicuous. The sample games as rendered by the first scheme, though consisting of digits and arithmetic signs, look no more like real arithmetic than the letters did—they're "arithmetic salad" at best. The second scheme, at first glance, looks better: at least the strings have the shape of equations. But, on closer examination, construed as equations, they would all be *false*—*wildly* false. In fact, though the signs are plausibly placed, the digits are just as randomly

"tossed" as the first case. The third scheme, by contrast, yields strings that not only look like equations, they *are* equations—they're all *true*. And this makes that third scheme seem much more acceptable. Why?

Consider a related problem: translating some ancient documents in a hitherto unknown script. Clearly, if some crank translator proposed a scheme according to which the texts came out gibberish (like the first one in the table) we would be unimpressed. Almost as obviously, we would be unimpressed if they came out *looking like* sentences, but *loony* ones: not just false, but scattered, silly falsehoods, unrelated to one another or to anything else. On the other hand, if some careful, systematic scheme finds in them detailed, sensible accounts of battles, technologies, facts of nature, or whatever, that we know about from other sources, then we will be convinced.[3] But again: why?

Translation is a species of interpretation (see p. 5 above). Instead of saying what some system thinks or is up to, a translator says what some strings of tokens (symbols) mean. To keep the two species distinct, we can call the former *intentional* interpretation, since it attributes intentional states, and the latter (translation) *semantic* interpretation, since it attributes meanings (= semantics).

Like all interpretation, translation is holistic: it is impossible to interpret a brief string completely out of context. For instance, the legal game 'HDJ A N' happens to come out looking just as true on the second as on the third scheme in our arithmetic example ('$2 \times 4 = 8$' and '$8 - 6 = 2$', respectively). But, in the case of the second scheme, this is obviously just an isolated coincidence, whereas, in the case of the third, it is part of a consistent pattern. Finding meaning in a body of symbols, like finding rationality in a body of behavior, is finding a certain kind of consistent, reliable *pattern*.

Well, what *kind* of pattern? Intentional interpretation seeks to construe a system or creature so that what it thinks and does turns out to be consistently reasonable and sensible, given its situation. Semantic interpretation seeks to construe a body of symbols so that what they mean ("say") turns out to be consistently reasonable and sensible, given the situation. This is *why* the third schemes in both the arithmetic and ancient-script examples are the acceptable ones: they're the ones that "make sense" of the texts, and *that's* the kind of pattern that translation seeks. I don't think we will ever have a precise, explicit definition of any phrase like "consistently reasonable and sensible, given the situation". But surely it captures much of what we mean (and Turing meant) by *intelligence*, whether in action or in expression.

3.2 Intelligence by explicit reasoning

Needless to say, interpretation and automation can be combined. A simple calculator, for instance, is essentially an automated version of the letter-game example, with the third interpretation. And the system that Turing envisioned—a computer with inputs and outputs that could be understood as coherent conversation in English—would be an interpreted automatic formal system. But it's *not* GOFAI.

So far, we have considered systems the inputs and outputs of which can be interpreted. But we have paid no attention to what goes on *inside* of those systems—*how* they get from an input to an appropriate output. In the case of a simple calculator, there's not much to it. But imagine a system that tackles harder problems—like "word problems" in an algebra or physics text, for instance. Here the challenge is not doing the calculations, but figuring out what calculations to do. There are many possible things to try, only one or a few of which will work.

A skilled problem solver, of course, will not try things at random, but will rely on experience and rules of thumb for guidance about what to try next, and about how things are going so far (whether it would be best to continue, to back-track, to start over, or even to give up). We can imagine someone muttering: "If only I could get that, then I could nail this down; but, in order to get that, I would need such and such. Now, let me see ... well, what if ..." (and so on). Such canny, methodical exploration—neither algorithmic nor random—is a familiar sort of articulate *reasoning* or *thinking* a problem out.

But each of those steps (conjectures, partial results, subgoals, blind alleys, and so on) is—from a formal point of view—just another token string. As such, they could easily be intermediate states in an interpreted automatic formal system that took a statement of the problem as input and gave a statement of the solution as output. Should these intermediate strings themselves then be *interpreted as* steps in thinking or reasoning the problem through? If two conditions are met, then the case becomes quite compelling. First, the system had better be able to handle with comparable facility an open-ended and varied range of problems, not just a few (the solutions to which might have been "pre-canned"). And, it had better be arriving at its solutions actually via these steps. (It would be a kind of fraud if it were really solving the problem in some other way, and then tacking on the "steps" for show afterwards.)

GOFAI is predicated on the idea that systems can be built to solve problems by reasoning or thinking them through in this way, and,

moreover, that this is how people solve problems. Of course, we aren't always consciously aware of such reasoning, especially for the countless routine problems—like those involved in talking, doing chores, and generally getting along—that we "solve" all the time. But the fact that we are not aware of it doesn't mean that it's not going on, subconsciously or somehow "behind the scenes".

The earliest GOFAI efforts emphasized problem-solving methods, especially the design of efficient heuristics and search procedures, for various specific classes of problems. (The article by Newell and Simon reviews this approach.) These early systems, however, tended to be quite "narrow-minded" and embarrassingly vulnerable to unexpected variations and oddities in the problems and information they were given. Though they could generate quite clever solutions to complicated problems that were carefully posed, they conspicuously lacked "common sense"—they were hopelessly *ignorant*—so they were prone to amusing blunders that no ordinary person would ever make.

Later designs have therefore emphasized broad, common-sense knowledge. Of course, problem-solving heuristics and search techniques are still essential; but, as research problems, these were overshadowed by the difficulties of large-scale "knowledge representation". The biggest problem turned out to be organization. Common-sense knowledge is vast; and, it seems, almost any odd bit of it can be just what is needed to avoid some dumb mistake at any particular moment. So all of it has to be at the system's "cognitive fingertips" all the time. Since repeated exhaustive search of the entire knowledge base would be quite impractical, some shortcuts had to be devised that would work most of the time. This is what efficient organizing or structuring of the knowledge is supposed to provide.

Knowledge-representation research, in contrast to heuristic problem solving, has tended to concentrate on natural language ability, since this is where the difficulties it addresses are most obvious. The principal challenge of ordinary conversation, from a designer's point of view, is that it is so often ambiguous and incomplete—mainly because speakers take so much for granted. That means that the system must be able to fill in all sorts of "trivial" gaps, in order to follow what's being said. But this is still GOFAI, because the filling in is being done rationally. Behind the scenes, the system is explicitly "figuring out" what the speaker must have meant, on the basis of what it knows about the world and the context. (The articles by Minsky and Dreyfus survey some of this work, and Dreyfus and Searle also criticize it.)

Despite its initial plausibility and promise, however, GOFAI has been in some ways disappointing. Expanding and organizing a system's store of explicit knowledge seems at best partially to solve the problem of common sense. This is why the Turing test will not soon be passed. Further, it is surprisingly difficult to design systems that can adjust their own knowledge in the light of experience. The problem is not that they can't modify themselves, but that it's hard to figure out just which modifications to make, while keeping everything else coherent. Finally, GOFAI systems tend to be rather poor at noticing unexpected similarities or adapting to unexpected peculiarities. Indeed, they are poor at recognizing patterns more generally—such as perceived faces, sounds, or kinds of objects—let alone *learning* to recognize them.

None of this means, of course, that the program is bankrupt. Rome was not built in a day. There is a great deal of active research, and new developments occur all the time. It *has* meant, however, that *some* cognitive scientists have begun to explore various alternative approaches.

4 New-fangled AI

By far the most prominent of these new-fangled ideas—we could call them collectively *NFAI* (*en*-fai)—falls under the general rubric of *connectionism.* This is a diverse and still rapidly evolving bundle of systems and proposals that seem, on the face of it, to address some of GOFAI's most glaring weaknesses. On the other hand, connectionist systems are not so good—at least not yet—at matching GOFAI's most obvious strengths. (This suggests, of course, a possibility of joining forces; but, at this point, it's too soon to tell whether any such thing could work, never mind how it might be done.) And, in the meantime, there are other NFAI ideas afloat, that are neither GOFAI nor connectionist. The field as a whole is in more ferment now than it has been since the earliest days, in the fifties.

4.1 Connectionist networks

Connectionist systems are networks of lots of simple active units that have lots of connections among them, by which they can interact. There is no central processor or controller, and also no separate memory or storage mechanism. The only activity in the system is these little units changing state, in response to signals coming in along those connections, and then sending out signals of their own. There are two ways in which such a network can achieve a kind of memory. First, in

the short term, information can be retained in the system over time insofar as the units tend to change state only slowly (and, perhaps, regularly). Second, and in the longer term, there is a kind of memory in the connections themselves. For, each connection always connects the same two units (they don't move around); and, more significant, each connection has a property, called its "weight" or "strength", which is preserved over time.

Obviously, connectionist networks are inspired to some extent by brains and neural networks. The active units are like individual neurons, and the connections among them are like the axons and dendrites along which electro-chemical "pulses" are sent from neuron to neuron. But, while this analogy is important, it should not be overstressed. What makes connectionist systems interesting as an approach to AI is not the fact that their structure mimics biology at a certain level of description, but rather what they can do. After all, there are countless other levels of description at which connectionist nets are utterly *un*biological; and, if some GOFAI account turns out to be right about human intelligence, then there will be *some* level of description at which it too accurately models the brain. Connectionist and allied research may someday show that neural networks are the level at which the brain implements psychological structures; but this certainly cannot be assumed at the outset.

In order to appreciate what is distinctive about network models, it is important to keep in mind how simple and relatively isolated the active units are. The "state" of such a unit is typically just a single quantitative magnitude—specifiable with a single number—called its *activation level*. This activation level changes in response to signals arriving from other units, but only in a very crude way. In the first place, it pays no attention to which signals came from which other units, or how any of those signals might be related to others: it simply adds them indiscriminately together and responds only to the total. Moreover, that response, the change in activation, is a simple function of that total; and the signal it then sends to other units is just a simple function of that resulting activation.

Now there is one small complication, which is the root of everything interesting about these models. The signal that a unit receives from another is not the same as the signal that the other unit sent: it is multiplied—increased or decreased—by the weight or strength of the connection between them. And there are always many more connections in a network than there are units, simply because each unit is

connected to many others. That means that the *overall* state of the network—that is, the *pattern* of activations of all its units—can change in very subtle and sophisticated ways, as a function of its initial state. The overall pattern of connection weights is what determines these complicated changes, and thus the basic character of the network.

Accordingly, connectionist networks are essentially *pattern processors*. And, it turns out, they can be quite good at certain psychologically important kinds of pattern processing. In particular, they are adept at finding various sorts of similarities among patterns, at recognizing repeated (or almost repeated) patterns, at filling in the missing parts of incomplete patterns, and at transforming patterns into others with which they have been associated. People are good at these kinds of pattern processing too; but GOFAI systems tend not to be, except in special cases. Needless to say, this is what gets cognitive scientists excited about connectionist models.

Two more points. First, when I say that networks are good at such pattern processing, I mean not only that they can do it well, but also that they can do it quickly. This is a consequence of the fact that, although each unit is very simple, there are a great many of them working at once—in *parallel*, so to speak—so the cumulative effect in each time increment can be quite substantial. Second, techniques have been discovered by means of which networks can be *trained* through exposure to examples. That is, the connection weights required for some desired pattern-processing ability can be induced ("taught") by giving the network a number of sample instances, and allowing it slowly to adjust itself. (It should be added, however, that the training techniques so far discovered are not psychologically realistic: people learn from examples too, but, for various reasons, we know it can't be in quite these ways.)

I mentioned a moment ago that GOFAI systems are not so good at pattern processing, except in special cases. In comparing approaches to mind design, however, it is crucial to recognize that some of these "special cases" are extremely important. In particular, GOFAI systems are remarkably *good* at processing (recognizing, transforming, producing) *syntactical* (grammatical) patterns of the sort that are characteristic of logical formulae, ordinary sentences, and many inferences. What's more, connectionist networks are *not* (so far?) particularly good at processing *these* patterns. Yet language is surely a central manifestation of (human) intelligence. No approach to mind design that cannot accommodate language ability can possibly be adequate.

Connectionist researchers use computers in their work just as much as GOFAI researchers do; but they use them differently. Pattern-processing networks are not themselves automatic formal systems: they do not manipulate formal tokens, and they are not essentially digital. To be sure, the individual units and connections are sharply distinct from one another; and, for convenience, their activations and weights are sometimes limited to a handful of discrete values. But these are more akin to the "digitization" of images in computer image processing than to the essential digitalness of chess pieces, logical symbols, and words. Thus, connectionist mind design relies on computers more in the way the weather service does, to simulate digitally systems that are not in themselves digital.

It has been shown, however, that some connectionist networks can, in effect, *implement* symbol manipulation systems. Although these implementations tend not to be very efficient, they are nevertheless interesting. For one thing, they may show how symbol manipulation could be implemented in the brain. For another, they might yield ways to build and understand genuine *hybrid* systems—that is, systems with the advantages of both approaches. Such possibilities aside, however, symbolic implementation would seem at best Phyrric victory: the network would be relegated to the role of "hardware", while the psychological relevance, the actual *mind design*, would still be GOFAI.

GOFAI is inspired by the idea that intelligence as such is made possible by explicit thinking or reasoning—that is, by the rational manipulation of internal symbol structures (interpreted formal tokens). Thus, GOFAI intentionality is grounded in the possibility of translation—*semantic* interpretation. Connectionist NFAI, by contrast, is inspired initially by the structure of the brain, but, more deeply, by the importance and ubiquity of non-formal pattern processing. Since there are no formal tokens (unless implemented at a higher level), there can be no semantically interpreted symbols. Thus, to regard these systems as having intentional states would be to adopt Dennett's intentional stance—that is, *intentional* interpretation.

In this volume, connectionist models are introduced and promoted in the articles by Rumelhart, by Smolensky, and by Churchland. The approach is criticized in the articles by Rosenberg and by Fodor and Pylyshyn. The articles by Ramsey, Stich and Garon and by Clark don't so much take sides as explore further what might be involved in the very idea of connectionism, in ways that might make a difference to those who do take sides.

4.2 Embodied and embedded AI

GOFAI is a fairly coherent research tradition, based on a single basic idea: thinking as internal symbol manipulation. 'NFAI', by contrast, is more a grab-bag term: it means, roughly, scientific mind design that is not GOFAI. Connectionism falls under this umbrella, but several other possibilities do as well, of which I will mention just one.

Connectionist and GOFAI systems, for all their differences, tend to have one feature in common: they accept an input from somewhere, they work on it for a while, and then they deliver an output. All the "action" is *within* the system, rather than being an integral part of a larger *interaction* with an active body and an active environment. The alternative, to put it radically (and perhaps a bit contentiously), would be to have the intelligent system *be* the larger interactive *whole*, including the body and environment as essential components. Now, of course, this whole couldn't be intelligent if it weren't for a special "subsystem" such as might be implemented in a computer or a brain; but, equally, perhaps, that subsystem couldn't be intelligent either except as part of a whole comprising the other components as well.

Why would anyone think this? It goes without saying that, in general, intelligent systems ought to be able to *act* intelligently "in" the world. That's what intelligence is for, ultimately. Yet, achieving even basic competence in real robots turns out to be surprisingly hard. A simple example can illustrate the point and also the change in perspective that motivates some recent research. Consider a system that must be able, among other things, to approach and unlock a door. How will it get the key in the lock? One approach would equip the robot with:

(1) precise sensors to identify and locate the lock, and monitor the angles of the joints in its own arm and hand;

(2) enough modelling power to convert joint information into a representation of the location and orientation of the key (in the coordinate system of the lock), compute the exact key motion required, and then convert that back into joint motions; and

(3) motors accurate enough to effect the computed motions, and thereby to slide the key in, smooth and straight, the first time.

Remarkably, such a system is utterly impractical, perhaps literally impossible, even with state-of-the-art technology. Yet insects, with far less compute power on board, routinely perform much harder tasks.

How would insectile "intelligence" approach the key-lock problem? First, the system would have a crude detector to notice and aim at

locks, more or less. But, it would generate no central representation of the lock's position, for other subsystems to use in computing arm movements. Rather, the arm itself would have its own ad hoc, but more local, detectors that enable it likewise to home in on a lock, more or less (and also, perhaps, to adjust its aim from one try to the next). And, in the meantime, the arm and its grip on the key would both be quite flexible, and the lock would have a kind of funnel around its opening, so any stab that's at all close would be guided physically right into the lock. Now *that's* engineering—elegant, cheap, reliable.

But is it *intelligence*? Well surely not much; but that may not be the right question to ask. Instead, we should wonder whether some similar essential involvement of the body (physical flexibility and special purpose subsystems, for instance) and the world (conveniences like the funnel) might be integral to capacities that are more plausibly intelligent. If so, it could greatly decrease the load on central knowledge, problem solving, and even pattern processing, thereby circumventing (perhaps) some of the bottlenecks that frustrate current designs.

To get a feel for the possibilities, move for a moment to the other end of the spectrum. Human intelligence is surely manifested in the ability to design and make things—using, as the case may be, boards and nails. Now, for such a design to work, it must be possible to drive nails into pieces of wood in a way that will hold them together. But neither a designer nor a carpenter ever needs to think about that—it need never even *occur* to them. (They take it for granted, as a fish does water.) The suitability of these materials and techniques is embedded in the structure of their culture: the logging industry, the manufacture of wire, the existence of lumber yards—and, of course, countless bodily skills and habits passed down from generation to generation.

Think how much "knowledge" is contained in the traditional shape and heft of a hammer, as well as in the muscles and reflexes acquired in learning to use it—though, again, no one need *ever* have thought of it. Multiply that by our food and hygiene practices, our manner of dress, the layout of buildings, cities, and farms. To be sure, some of this was explicitly figured out, at least once upon a time; but a lot of it wasn't—it just evolved that way (because it worked). Yet a great deal, perhaps even the bulk, of the basic expertise that makes human intelligence what it is, is maintained and brought to bear in these "physical" structures. It is neither stored nor used inside the head of *anyone*—it's in their bodies and, even more, out there in the world.

Scientific research into the kinds of systems that might achieve intelligence in this way—embodied and embedded mind design—is still in an early phase. Two rather different theoretical and empirical strategies are presented here in the articles by Brooks and van Gelder.

5 What's missing from mind design?

A common complaint about artificial intelligence, of whatever stripe, is that it pays scant attention to feelings, emotions, ego, imagination, moods, consciousness—the whole "phenomenology" of an inner life. No matter how smart the machines become, so the worry goes, there's still "nobody home". I think there is considerable merit in these misgivings, though, of course, more in some forms than in others. Here, however, I would like briefly to discuss only one form of the worry, one that strikes me as more basic than the others, and also more intimately connected with cognition narrowly conceived.

No current approach to artificial intelligence takes *understanding* seriously—where understanding itself is understood as distinct from knowledge (in whole or in part) and prerequisite thereto. It seems to me that, taken in this sense, *only people* ever understand anything—no animals and no artifacts (yet). It follows that, in a strict and proper sense, no animal or machine genuinely believes or desires anything either—How could it believe something it doesn't understand?—though, obviously, in some other, weaker sense, animals (at least) have plenty of beliefs and desires. This conviction, I should add, is not based on any in-principle barrier; it's just an empirical observation about what happens to be the case at the moment, so far as we can tell.

So, what is it for a system to understand something? Imagine a system that makes or marks a battery of related distinctions in the course of coping with some range of objects. These distinctions can show up in the form of differing skillful responses, different symbol structures, or whatever. Let's say that, for each such distinction, the system has a *proto-concept*. Now I suggest that a system *understands* the objects to which it applies its proto-concepts insofar as:

(1) it takes responsibility for applying the proto-concepts correctly;

(2) it takes responsibility for the empirical adequacy of the proto-concepts themselves; and

(3) it takes a firm stand on what can and cannot happen in the world, when grasped in terms of these proto-concepts.

When these conditions are met, moreover, the proto-concepts are not merely *proto*-concepts, but *concepts* in the full and proper sense.

The three conditions are not unrelated. For, it is precisely in the face of something *impossible* seeming to have happened, that the question of *correct* application becomes urgent. We can imagine the system responding in some way that we would express by saying: "This *can't* be right!" and then trying to figure out what went wrong. The responsibility for the concepts themselves emerges when, too often, it can't find any mistake. In that event, the conceptual structure itself must be revised, either by modifying the discriminative abilities that embody the concepts, or by modifying the stand it takes on what is and isn't possible, or both. Afterward, it will have (more or less) new concepts.

A system that appropriates and takes charge of its own conceptual resources in this way is not merely going through the motions of intelligence, whether evolved, learned, or programmed-in, but rather grasps the point of them for itself. It does not merely make discriminations or produce outputs that, when best interpreted by us, come out true. Rather, such a system appreciates for itself the difference between truth and falsity, appreciates that, in these, it must accede to the world, that the world determines which is which—and it *cares*. That, I think, is *understanding*.[4]

Notes

1. Both parts of this idea have their roots in W. V. O. Quine's pioneering (1950, 1960) investigations of meaning. (Meaning is the linguistic or symbolic counterpart of intentionality.)
2. Chess players will know that the rules for castling, stalemate, and capturing *en passent* depend also on *previous* events; so, to make chess strictly formal, these conditions would have to be encoded in further tokens (markers, say) that count as part of the current position.
3. A similar point can be made about code-cracking (which is basically translating texts that are contrived to make that especially difficult). A cryptographer knows she has succeeded when and only when the decoded messages come out consistently sensible, relevant, and true.
4. These ideas are explored fruther in the last four chapters of Haugeland (1997).

Computing Machinery and Intelligence

2

A. M. Turing
1950

1 The imitation game

I propose to consider the question "Can machines think?" This should begin with definitions of the meaning of the terms 'machine' and 'think'. The definitions might be framed so as to reflect so far as possible the normal use of the words, but this attitude is dangerous. If the meaning of the words 'machine' and 'think' are to be found by examining how they are commonly used it is difficult to escape the conclusion that the meaning and the answer to the question, "Can machines think?" is to be sought in a statistical survey such as a Gallup poll. But this is absurd. Instead of attempting such a definition I shall replace the question by another, which is closely related to it and is expressed in relatively unambiguous words.

The new form of the problem can be described in terms of a game which we call the "imitation game". It is played with three people, a man (A), a woman (B), and an interrogator (C) who may be of either sex. The interrogator stays in a room apart from the other two. The object of the game for the interrogator is to determine which of the other two is the man and which is the woman. He knows them by labels X and Y, and at the end of the game he says either "X is A and Y is B" or "X is B and Y is A". The interrogator is allowed to put questions to A and B thus:

C: Will X please tell me the length of his or her hair?

Now suppose X is actually A, then A must answer. It is A's object in the game to try to cause C to make the wrong identification. His answer might therefore be

A: My hair is shingled, and the longest strands are about nine inches long.