Republican Automatons (1920) by George Grosz.
Watercolor, 23 5/8 x 18 5/8". Collection, The Museum of Modern Art, New York, Advisory Committee Fund.

COMPUTER POWER AND HUMAN REASON

FROM JUDGMENT TO CALCULATION

Joseph Weizenbaum

THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY

W. H. FREEMAN AND COMPANY
New York

5

THEORIES AND MODELS

Suppose a team of explorers from a highly technological society just like ours, but one that knew nothing about computers, were to come upon a functioning computer. They find that they cannot break into it, can gain no access to its, so to say, electro-neurophysiological apparatus. They do notice, however, that whenever they type something on its console typewriter, the computer's lights flash in a complex but apparently orderly way, its magnetic tapes sometimes spin, and the typewriter types a message that appears to be a response to what they have typed. After a time, they discover that they can dismount the computer's magnetic tapes and cause their contents to be printed on another device, a high-speed printer, which they also find on the site of their exploration. These contents prove to be readable, at least in the sense that they are represented in the explorer's own alphabet.

Since this machine—and the explorers do recognize it as a machine—is obviously a behaving instrument, the explorers naturally wish to discover the laws of its behavior. How could they go about reaching the understanding they desire? Indeed, what can it mean to understand the machine's laws of behavior?

We, of course, can put ourselves in the position of a highly privileged observer, somewhat like that of a chemistry instructor who knows very well what compound he gave his students to analyze. We know that the machine the explorers found is a computer, moreover, a computer of precisely such and such a type and one containing a particular program we also know in detail. We can therefore tell the precise degree, so to say, of understanding the explorers will have achieved at any given stage of their research. If, for example, they were to produce a computer of their own which, as seen from our privileged perspective, appears to be an exact copy of the computer they found and which even contains the same program as the original, then we would have to say that they understood the original computer at least as well as did its designers.

However, lesser achievements would also deserve to be called understanding of a very high degree. Suppose, for example, that the explorers managed to build a computer whose internal structure and whose internal components are entirely different, but whose input-output behavior is indistinguishable from that of the original; in other words, no test short of breaking open either computer can determine which of the two computers, the one the explorers found or the one they built, generated what response to what input. It may be that the internal components of the found machine are made of bailing wire, chewing gum, and adhesive tape, whereas those of the explorers' functional copy are all electronic; that doesn't matter as long as, for any reason, the original machine may not be opened for detailed internal inspection. (Actually, of course, such an achievement is impossible in principle. It may be, for example, that the original computer was so constructed that it prefaces its first console typewriter output with an exclamation mark on and only on the seventeenth Thursday of every leapyear. Even if that were discovered, the explorers could never be sure that there are not other oddities which, though systematic, have not yet been discovered.

We, as privileged observers, would, of course, know about such things.)

A still lesser degree of understanding could be claimed if the explorers succeeded in building some sort of digital computer, say, a simple universal Turing machine of the type we discussed in Chapter II, and then explained the machine they found in terms of Turing-machine principles. They could then account for the found machine's extraordinary versatility and even for the fact, say, that it takes it longer to compute the inverse of a large matrix than that of a smaller one. They might, on the other hand, be utterly unable to explain why it takes the found computer longer to execute algorithms given to it in one programming language than it does to execute those same algorithms written in another programming language. We, given our omniscience about computers, know that the difference in execution speeds is due to the fact that the computer translates programs in the first language a line at a time into its machine language, and then obeys the so-generated machine-language instructions, whereas it first translates the whole program written in the second language into its machine language and only then executes the entire set of so-generated machine-language instructions. The latter process is almost always much less time-consuming than the former. Presumably the explorers would eventually develop some explanation for this and other puzzling phenomena. They might, for example, conjecture that some programming languages are more familiar to the machine than some others, and might even develop some taxonomy of programming languages based on the machine's experimentally discovered familiarity with them. The concept of familiarity, as well as the taxonomy of languages for which it serves as an organizing principle, would then become part of their computer science. It is, of course, a concept weak in explanatory power, even a misleading one. But then it is much easier for us privileged observers to know this than it is for the explorers, faced as they are with the task of having to explain phenomena of unbounded complexity.

Let us press this fantasy one more step: Suppose the explorers found not just one computer, but many computers of many diverse types, all of them, however, so-called single-address machines.

Recall that a single-address machine is one whose built-in instructions have the form "operation code; address of datum to be operated on" (see p. 86). With luck, and if the explorers were clever, they would discover what they would undoubtedly call a "language universal" with respect to the grammatical structure of the machine languages they have encountered. And to explain it as something other than a mere accident (which would, of course, be no explanation at all), they would have to conclude that this universal feature of all the languages they have observed must be due to some correspondingly universal feature, some innate property, of the machines themselves. And they would, as we privileged observers know, of course, be correct; the fact that a computer's machine language has the single-address format is a direct consequence of its design. Indeed, if we assume that the machines the explorers discovered are ordinary computers and not robots—that is, that they don't have perceptors, like television eyes, and effectors, like mechanical arms and hands—then all the discoveries the explorers make and all the theories they develop must be based solely on observations of the, so to say, verbal behavior of the machines. Apart from such minor, though possibly not unhelpful, phenomena as the flashing of the computers' lights and the occasional motions of their tape reels, the only evidence of their structures that the computers provide is, after all, linguistic. They accept strings of linguistic inputs in the form of the texts typed on their console typewriters, and they respond with linguistic outputs written on the same instrument or onto magnetic tapes.

In Chapters II and III we were very much concerned with legal moves in abstract games and grammatical constructions in abstract languages. My aim there was to build up the idea of a computer on the basis of such concepts. In the fantasy we are currently entertaining, we are, in effect, looking at the other side of the same coin. We now see that, if we strive to explain computers when bounded by the restriction that we may not break the computer open, then all explanations must be derived from linguistic bases.

The position of a human being observing another human being is not so very different from that of the explorers who wish to understand the computers they have encountered. We too have ex-

tremely limited access to the neurophysiological material that appears to determine how we think. Besides, it wouldn't advance our current understanding of thinking very much even if we could subject the living brain to the kind of analysis to which we actually can subject a running computer, that is, by tracing connections, electrical pulses, and so on. Our ignorance of brain function is currently so very nearly total that we could not even begin to frame appropriate research strategies. We would stand before the open brain, fancy instruments in hand, roughly as an unschooled laborer might stand before the exposed wiring of a computer: awed perhaps, but surely helpless. A microanalysis of brain functions is, moreover, no more useful for understanding anything about thinking than a corresponding analysis of the pulses flowing through a computer would be for understanding what program the computer is running. Such analyses would simply be at the wrong conceptual level. They might help to decide crucial experiments, but only after such experiments had been designed on the basis of much higher-level (for example, linguistic) theories.

Because, in fact, scientists do suffer from the same sort of handicaps as we imposed on our mythical explorers, and cannot communicate with an omniscient observer who could, if he but would, reveal all secrets, it is not surprising that at least some scientists seek understanding of the way humans work in somewhat the same way as our explorers might have sought to understand the computers they found, that is, by designing computers whose imput-output behavior resembles that of humans as closely as possible.

The work of linguists—for example, that of Noam Chomsky—should be mentioned here, even though it does not involve the use of computers. A simpleminded and grossly misleading view of the task that Chomsky's school has set itself is that it is to systematically record the grammatical rules of as many natural languages (e.g., English) as possible. If that were the only, or even the principal, aim of Chomsky's school, we would expect it to publish a series of books, all independent of one another, entitled *"The Grammar of X,"* where *X* stands for one of the various known human languages. In fact, Chomsky's most profoundly significant working hypothesis is that man's genetic endowment gives him a set of highly specialized abilities and imposes on him a corresponding set of restrictions which, taken together, determine the number and the kinds of degrees of freedom that govern and delimit all human language development.

To understand how a "specialized ability" may simultaneously be a "corresponding restriction," we need only remind ourselves of a single-address machine. The fact that such a machine can decode a machine-language instruction in terms of a component (say, its eight leftmost bits) that is the instruction's operation code, and another component (its remaining bits) that is its address portion, implies at once that no other instruction format is, for that machine, admissible. Indeed, the very idea of the grammaticality of a whole computer program, let alone that of a single machine-language instruction, implies that there exist some symbol strings which, while they may superficially look like programs, are unintelligible and hence not admissible as programs. What is seen from one point of view as a specialized ability of a machine must be seen as a restriction from another perspective.

How then, Chomsky asks, can we gain an insight into the genetically endowed abilities that we call the mind? He answers that, given our present state of virtually total ignorance about the living brain, our best chance is to infer the innate properties of the mind from the highly restrictive principles of a "universal grammar." The linguist's first task is therefore to write grammars, that is, sets of rules, of particular languages, grammars capable of characterizing all and only the grammatically admissible sentences of those languages, and then to postulate principles from which crucial features of all such grammars can be deduced. That set of principles would then constitute a universal grammar. Chomsky's hypothesis is, to put it another way, that the rules of such a universal grammar would constitute a kind of projective description of important aspects of the human mind. He does not believe, of course, that people know these rules in the same way that they know, say, the rules of long division. Instead they know them (to use Polanyi's word) tacitly, that is, in the same way that people know how to maintain their balance while running. In both speaking and running, by the way, performance

pinges on him from his environment, and that his actions, especially his verbal behavior, inform his environment in turn. Whatever else man is, then, and again he is very much else, he is also a receiver and a transmitter of information. But even so, he is certainly more than a mere mirror that reflects more or less precisely whatever signals impinge on it; for he attends to only a small fraction of what William James called "the blooming, buzzing confusion" of sensations with which his environment bombards him, and he transforms that distillate of his world into memories, mental imagery of many sorts, speech and writing, strokes on piano keyboards, in short, into thought and behavior. Whatever else man is, then, and he is much else, he is also an information processor.

I will, in what follows, try to maintain the position that there is nothing wrong with viewing man as an information processor (or indeed as anything else) nor with attempting to understand him from that perspective, providing, however, that we never act as though any single perspective can comprehend the whole man. Seeing man as an information-processing system does not in itself dehumanize him, and may very well contribute to his humanity in that it may lead him to a deeper understanding of one specific aspect of his human nature. It could, for example, be enormously important for man's understanding his spirituality to know the limits of the explanatory power of an information-processing theory of man. In order for us to know those limits, the theory would, of course, have to be worked out in considerable detail.

Before we discuss what an information-processing theory of man might look like, I must say more about theories and especially about their relation to models. A theory is first of all a text, hence a concatenation of the symbols of some alphabet. But it is a symbolic construction in a deeper sense as well; the very terms that a theory employs are symbols which, to paraphrase Abraham Kaplan, grope for their denotation in the real world or else cease to be symbolic.[3] The words "grope for" are Kaplan's, and are a happy choice—for to say that symbols "find" their denotation in the real world would deny, or at least obscure, the fact that the symbolic terms of a theory can never be finally grounded in reality.

Definitions that define words in terms of other words leave those other words to be defined. In science generally, symbols are often defined in terms of operations. In physics, for example, mass is, informally speaking, that property of an object which determines its motion during collision with other objects. (If two objects moving at identical velocities come to rest when brought into head-on collision, it is said that they have the same mass.) This definition of mass permits us to design experiments involving certain operations whose outcomes "measure" the mass of objects. Momentum is defined as the product of the mass of an object and its velocity (mv), acceleration as the rate of change of velocity with time ($a = dv/dt$), and finally force as the product of mass and acceleration ($f = ma$). In a way it is wrong to say that force is "defined" by the equation $f = ma$. A more suitable definition given in some physics texts is that force is any influence capable of producing a change in the motion of a body.[4] The difference between the two senses of "definition" alluded to here illustrates that so-called operational definitions of a theory's terms provide a basis for the design of experiments and the discovery of general laws, but that these laws may then serve as implicit definitions of the terms occurring in them. These and still other problematic aspects of definition imply that all theoretic terms, hence all theories, must always be characterized by a certain openness. No term of a theory can ever be fully and finally understood. Indeed, to once more paraphrase Kaplan, it may not be possible to fix the content of a single concept or term in a sufficiently rich theory (about, say, human cognition) without assessing the truth of the whole theory.[5] This fact is of the greatest importance for any assessment of computer models of complex phenomena.

A theory is, of course, not merely any grammatically correct text that uses a set of terms somehow symbolically related to reality. It is a systematic aggregate of statements of laws. Its content, its very value as theory, lies at least as much in the structure of the interconnections that relate its laws to one another, as in the laws themselves. (Students sometimes prepare themselves for examinations in physics by memorizing lists of equations. They may well pass their examinations with the aid of such feats of memory, but it can hardly

model of B if that theory of B is a theory of A as well. We accept the condition also mentioned by Kaplan that there must be no causal connection between the model and the thing modelled; for if a model is to be used as an explanatory tool, then we must always be sure that any lessons we learn about a modeled entity by studying its model would still be valid if the model were removed.

People do, of course, derive consequences from theories without building explicit models like, say, scaled-down wings in wind tunnels. But that is not to say that they derive such consequences without building models at all. When a psychiatrist applies psychoanalytic theory to data supplied to him by his patient, he is, so to speak, exercising a mental model, perhaps a very intuitive one, of his patient, a model cast in psychoanalytic terms. To state it one way, the analyst finds the study of his mental model (A) of his patient (B) useful for understanding his patient (B). To state it another way, the analyst believes that psychoanalytic theory applies to his patient and therefore constructs a model of him in psychoanalytic terms, a model to which, of course, psychoanalytic theory also applies. He then transforms (translates is perhaps a better word) inferences derived from working with the model into inferences about the patient. (It has to be added, lest there be a misunderstanding, that however much the practicing psychoanalyst is committed to psychoanalytic theory and however much his attitudes are shaped by it, psychoanalytic therapy consists in only small part of direct or formal application of theory. Nevertheless, it is plausible that all of us make all our inferences about reality from mental models whose structures, and to a large extent whose contents as well, are strongly determined by our explicitly and implicitly held theories of the world.)

Computers make possible an entirely new relationship between theories and models. I have already said that theories are texts. Texts are written in a language. Computer languages are languages too, and theories may be written in them. Indeed, for the present purpose we need not restrict our attention to machine languages or even to the kinds of "higher-level" languages we have discussed. We may include all languages, specifically also natural languages, that computers may be able to interpret. The point is

precisely that computers do *interpret* texts given to them, in other words, that texts determine computers' behavior. Theories written in the form of computer programs are ordinary theories as seen from one point of view. A physicist may, for example, communicate his theory of the pendulum either as a set of mathematical equations or as a computer program. In either case he will have to identify the terms of his theory—his "variables," in technical jargon—with whatever they are to correspond to in reality. (He may say l is the length of the pendulum's string, p its period of oscillation, g the acceleration due to gravity, and so on.) But the computer program has the advantage not only that it may be understood by anyone suitably trained in its language, just as a mathematical formulation can be readily understood by a physicist, but that it may also be run on a computer. Were it to be run with suitable assignments of values to its terms, the computer would *simulate* an actual pendulum. And inferences could be drawn from that simulation, and could be directly translated into inferences applicable to real pendulums. A theory written in the form of a computer program is thus both a theory and, when placed on a computer and run, a model to which the theory applies. Newell and Simon say about their information-processing theory of human problemsolving, "the theory performs the tasks it explains."[6] Strictly speaking, a theory cannot "perform" anything. But a model can, and therein lies the sense of their statement. We shall, however, have to return to the troublesome question of what the performance of a task can and cannot explain.

In order to aid our intuition about what it means for a computer model to "behave," let us briefly examine an exceedingly simple model: We know from physics, and indeed it follows from the equation $f = ma$ that we mentioned earlier, that the distance d an object will fall in a time t is given by

$$d = at^2/2,$$

where a is the acceleration due to gravity. In most elementary physics texts, a is simply asserted to be the earth's gravitational constant, namely, 32 ft/sec^2, where the unit of distance is feet and that of time is seconds. The equation itself is a simple mathematical model of a

falling object. If we assume, for the sake of simplicity, that the acceleration a is indeed constant, namely, 32 ft/sec^2, we can compute how far an object will have fallen after, say, 4 seconds: $4 \times 4 = 16$ and $16 \times 32 = 512$ and $512 + 2 = 256$. The answer, as schoolchildren would say, is therefore 256 feet.

Mathematicians long ago fell into the habit of writing the so-called variables that appear in their equations as single letters. Perhaps they did this to guard against writer's cramp or to save chalk. Whatever their reasons, their notation is somewhat less than maximally mnemonic. Because computer programs are often intended to be read and understood by people, as well as to be executed by computers, and since computers are, within limits, indifferent to the lengths of the symbol strings they manipulate, computer programmers often use whole words to denote the variables that appear in their programs. Other considerations make it inconvenient to use juxtaposition of variables, as in xy, to indicate multiplication. Instead the symbol "*" is used in many programming languages. Similarly, "**" is used to indicate exponentiation. Thus, where the mathematician writes t^2, the programmer writes t**2. The equation

$$d = at^2/2$$

when transformed into a program statement* may thus appear as

distance = (acceleration * time **2)/2.

Let us now complicate our example just a little. Suppose an object is to be dropped from a stationary platform, say, a helicopter

* A significant technical point must be made here. Although the "statement" shown here is a transliteration of the equation to which it corresponds, it is not itself an equation. In technical parlance, it is an "assignment statement." It assigns a value to the variable "distance." "Distance," in turn, is technically an "identifier," the name of a storage location in which is stored the value which has been assigned to the corresponding variable. In mathematics, a variable is an entity whose value is not known, but which has a definite value nonetheless, a value that can be discovered by solving the equation. In programs, a variable may have different values at different stages of the execution of the program. In ordinary mathematics, e.g., in high-school algebra, the "equation" "$x = x + 1$" is nonsense. The same string of symbols appearing as an expression in a program has meaning, namely, that 1 is to be added to the contents of the location denoted by "x" and those contents replaced by the resulting sum.

hovering at some altitude above the ground. The object's height above the ground after it has fallen for some time would then be given by

height = altitude − (acceleration * time ** 2)/2.

Finally, suppose that the helicopter is flying forward at some constant velocity while maintaining its altitude. If there were no aerodynamic effects on the object dropped from the helicopter, it would remain exactly below the helicopter during its entire journey to the ground. The object's horizontal displacement from the point over which it was dropped would therefore be the same as the helicopter's horizontal displacement from that point, that is,

displacement = velocity * time,

where by "velocity" we here, of course, mean the helicopter's velocity.

We now have, from one point of view, two equations, from another point of view, two program statements, from which we can compute the horizontal and vertical coordinates of an object dropped from a moving helicopter. We can combine them and imbed them in a small fragment of a computer program, as follows:

```
FOR time = 0 STEP .001 UNTIL height = 0 DO;
    height = altitude − (acceleration * time**2) / 2 ;
    displacement = velocity * time ;
    display (height, displacement) ;
END.
```

This is an example of a so-called *iteration statement.* It tells the computer to do a certain thing until some condition is achieved. In this case, it tells the computer to first set the variable "time" to zero, then to compute the height and displacement of what we would interpret to be the falling object, then to display the coordinates so computed—I shall say more about displaying in a moment—and, if the computed height is not zero, to add .001 to the variable "time"

and do the whole thing again, that is, to iterate the process. (This program contains an error which, for the sake of simplicity, I have let stand. As it is, it may run forever. To repair it, the expression "height = 0" should be replaced by "height < 0." The reason for this is left to the reader to discover.)

We have assumed here that the computer on which this program is to run has a built-in display apparatus and the corresponding display instruction. We may imagine the computer's display to be a cathode-ray tube like that of an ordinary television set. The display instruction delivers two numbers to this device, in this example, the values of height and displacement. The display causes a point of light to appear on its screen at the place whose coordinates are determined by these two numbers, i.e., so many inches up and so many inches to the right of some fixed point of origin.

If we now make some additional assumptions about for example, the persistence of the lighted dot on the screen and the overall timing of the whole affair, we can imagine that the moving dot we see will appear to us like a film of the object falling from the helicopter (see Figure 5.1). It is thus possible, even compelling, to think of the computer "behaving," and for us to interpret its behavior as modeling that of the falling object.

It would be very easy for us to complicate our example step by step, first, for example, by extending it to cover the trajectory of a missile fired from a gun and, with that as a base, to extend it to the flight of orbiting satellites. We would then have described at least

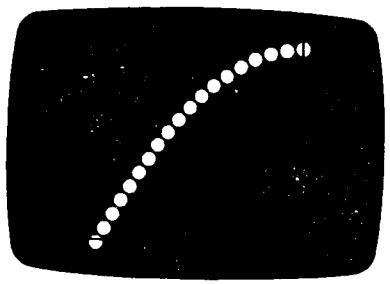

Figure 5.1.
Cathode simulation of the trajectory of an object dropped from a flying helicopter.

the most fundamental basis on which the orbital simulations we often see on television are developed. But that is not my purpose. Simple as our example is, we can learn pertinent lessons from it.

To actually use the model, an investigator would initialize it by assigning values to the parameters altitude and velocity, run it on an appropriate computer, and observe its behavior on the computer's display device. There would, however, be discrepancies between what the model, so to speak, says a falling object would do and the behavior of its real counterpart. The model, for example, makes the implicit assumption that there are no aerodynamic effects on the falling object. But we know that there would certainly be air resistance in the real situation. Indeed, if the object dropped were a parachute, its passenger's life would depend on air resistance slowing its fall. A model is always a simplification, a kind of idealization of what it is intended to model.

The aim of a model is, of course, precisely not to reproduce reality in all its complexity. It is rather to capture in a vivid, often formal, way what is essential to understanding some aspect of its structure or behavior. The word "essential" as used in the above sentence is enormously significant, not to say problematical. It implies, first of all, purpose. In our example, we seek to understand how the object falls, and not, say, how it reflects sunlight in its descent or how deep a hole it would dig on impact if dropped from such and such a height. Were we interested in the latter, we would have to concern ourselves with the object's weight, its terminal velocity, and so on. We select, for inclusion in our model, those features of reality that we consider to be essential to our purpose. In complex situations like, say, modeling the growth, decay, and possible regeneration of a city, the very act of choosing what is essential and what is not must be at least in part an act of judgment, often political and cultural judgment. And that act must then necessarily be based on the modeler's intuitive mental model. Testing a model may reveal that something essential was left out of it. But again, judgment must be exercised to decide what the something might be, and whether it is "essential" for the purpose the model is intended to serve. The ultimate criteria, being based on intentions and pur-

poses as they must be, are finally determined by the individual, that is, human, modeler.

The problem associated with the question of what is and what is not "essential" cuts the other way as well. A model is, after all, a different object from what it models. It therefore has properties not shared by its counterpart. The explorers we mentioned earlier may have built a functional model of the computer they found by using light-carrying fibers and light valves, whereas the real computer used wires and the kind of electronic gates we considered in Chapter III. They could then easily have come to believe that light is essential to the operation of computers. Their computer science might have included large elements of physical optics, and so on. It is indeed possible to build computers using light-carrying fibers, etc. Their logical diagrams, that is, their paper designs, would, up to a point, be indistinguishable from those of the corresponding electronic computers, because the former would have the same structure as the latter. What is essential about a computer is the organization of its components and not, again up to a point, precisely what those components are made of. Another example: there are people who believe it possible to build a computer model of the human brain on the neurological level. Such a model would, of course, be in principle describable in strictly mathematical terms. This might lead some people to believe that the language our nervous system uses must be the language of our mathematics. Such a belief would be an error of the kind we mean. John von Neumann, the great computer pioneer, touched briefly on this point himself:

> "When we talk mathematics, we may be discussing a *secondary* language, built on the *primary* language truly used by the central nervous system. Thus the outward forms of *our* mathematics are not absolutely relevant from the point of view of evaluating what the mathematical or logical language *truly* used by the central nervous system is."[7]

One function of a model is to test theories at their extreme limits. I have already mentioned that computers can generate films that model the behavior of a particle at extreme limits of relativistic

velocities. Our own simple model of falling objects could be used in its present form to simulate, hence to calculate, the fall of an object from a spaceship flying near the surface of the moon. All we would have to do is to initialize acceleration to the number appropriate for the gravity existing on the moon's surface (providing, of course, that the spaceship is not so high above the surface of the moon that the effect of the moon's gravitational field would have been significantly changed—another implicit assumption). For that simulation exercise we would not have to have any components in our model corresponding to air resistance or other aerodynamic effects: the moon has no atmosphere. (Recall that an astronaut simultaneously dropped a feather and a hammer onto the moon's surface and that they both reached the ground at the same time.)

It is a fact, however, that the moon's gravitational field varies from place to place. These variations are thought to be due to so-called masscons, that is, concentrations of mass within the moon that act somewhat like huge magnets irregularly buried deep within the moon. The masscon hypothesis was advanced to account for observed irregularities in the trajectories of spacecraft orbiting the moon. It is, in effect, an elaboration of the falling-body model we have discussed. The elaborated model is the result of substituting a complex mathematical function (in other words, a subroutine) for the single term "acceleration" of our simple model. I mention it to illustrate the process, in this case properly applied, of elaborating a model to account for new and unanticipated observations. But the masscon elaboration was not the only possible extension of either the theory or its computer model. It could have been hypothesized, for example, that the moon is surrounded by a turbulent ether mantle whose waves and eddies caused the spaceship's irregular behavior. There are dozens of very good reasons for rejecting this hypothesis, of course, but a good programmer, given a lot of data, could more or less easily elaborate the model with which we started by adding "ether turbulence subroutines" so that, in the end, the model behaved just as the spaceship was observed to behave. Such a model would, of course, no longer look simple. Indeed, its very complexity, plus the precision to which it carried its calculations, might lend it a certain credibility.

Earlier I said that the value of a theory lies not so much in the aggregation of the laws it states as in the structure that interconnects them. The trouble with the kind of model elaboration that would result from such an "ether turbulence" hypothesis is that it simply patches one more "explanation" onto an already existing structure. It is a patch in that it has no roots in anything already present in the structure. Computer models have, as we have seen, some advantages over theories stated in natural language. But the latter have the advantage that patching is hard to conceal. If a theory written in natural language is, in fact, a set of patches and patches on patches, its lack of structure will be evident in its very composition. Although a computer program similarly constructed may reveal its impoverished structure to a trained reader, this kind of fault cannot be so easily seen in the program's performance. A program's performance, therefore, does not alone constitute an adequate validation of it as theory.

I have already alluded to the heuristic function of theories. Since models in computer-program form are also theories (at least, some programs deserve to be so thought of), what I have said about theories in general also applies to them, perhaps even more strongly, in this sense: in order for us to draw consequences from discursive theories, even to determine their coherence and consistency, they must, as I have said, be modeled anyway, that is, be modeled in the mind. The very eloquence of their statements, especially in the eyes of their authors, may give them a persuasive power they hardly deserve. Besides, much time may elapse between the formulation of a theory and its testing in the minds of men. Computer programs tend to reveal their errors, especially their lack of consistency, quickly and sharply. And, in skilled hands, computer modeling provides a quick feedback that can have a truly therapeutic effect precisely because of its immediacy. Computer modeling is thus somewhat like Polaroid photography: it is hard to maintain the belief that one has taken a great photograph when the counterexample is in one's hands. As Patrick Suppes remarked,

> The attempt to characterize exactly models of an empirical theory almost inevitably yields a more precise and clearer understanding of the exact character of a theory. The emptiness and shallowness of many classical theories in the social sciences is well brought out by the attempt to formulate in any exact fashion what consitutes a model of the theory. The kind of theory which mainly consists of insightful remarks and heuristic slogans will not be amenable to this treatment. The effort to make it exact will at the same time reveal the weakness of the theory."[8]

The question is, of course, just what kinds of theories are "amenable to this treatment?"

6

COMPUTER MODELS IN PSYCHOLOGY

Sometimes a very complex idea enters the public consciousness in a form so highly simplified that it is little more than a caricature of the original; yet this mere sketch of the original idea may nevertheless change the popular conception of reality dramatically. For example, consider Einstein's theory of relativity. Just how and why this highly abstract mathematical theory attracted the attention of the general public at all, let alone why it became for a time virtually a public mania and its author a pop-culture hero, will probably never be understood. But the same public which clung to the myth that only five people in the world could understand the theory, and which thus acknowledged its awe of it, also saw the theory as providing a new basis for cultural pluralism; after all, science had now established that everything is relative. A more recent example may be found in the popular reception of the work of F. Crick and J. D.

Watson, who shared the Nobel prize in Medicine in 1962 for their studies of the molecular structure of DNA, the nucleic acid within the living cell that transmits the hereditary pattern. Here again highly technical results, reported in a language not at all comprehensible to the layman, were grossly oversimplified and overgeneralized in the public mind into the now-popular inpression that it is already possible to design a human being to specifications decided on in advance. In one fell swoop, the general public created for itself a vision of a positive eugenics based not on such primitive and (I hope) abhorrent techniques as the killing and sterilization of "defectives," but on the creation of supermen by technological means. What these two examples have in common is that both have introduced new metaphors into the common wisdom.

A metaphor is, in the words of I. A. Richards, "fundamentally a borrowing between and intercourse of thoughts, a transaction between contexts."[1] Often the heuristic value of a metaphor is not that it expresses a new idea, which it may or may not do, but that it encourages the transfer of insights, derived from one of its contexts, into its other context. Its function thus closely resembles that of a model. A Western student of Asian societies may, for example, not learn anything directly from the metaphoric observation that the overseas Chinese are the Jews of Asia. But it may never have occurred to him that the position of Jews in the Western world, e.g., as entrepreneurs, intellectuals, and targets of persecution, may serve as a model that can provoke insights and questions relevant for understanding the social role and function of, say, the Chinese in Indonesia. Although calling that possibility to his attention may not give the Western student a new idea, it may enable him to derive new ideas from the interchange of two contexts, neither of which are themselves new to him, but which he had never before connected.

Neither the idea of one object moving relative to another, nor that of man being fundamentally a physical object, was new to the common wisdom in the 1920's and the 1960's, respectively. What struck the popular imagination when, for some reason, the press campaigned for Einstein's theory, was that science appeared to have pronounced relativity to be a fundamental and universal fact. Hence the slogan "everything is relative" was converted into a legiti-

mate contextual framework which could, potentially at least, be coupled to every other universe of discourse, e.g., as an explanatory model legitimating cultural pluralism.

The results announced by Crick and Watson fell on a soil already prepared by the public's vague understanding of computers, computer circuitry, and information theory (with its emphasis on codes and coding), and, of course, by its somewhat more accurate understanding of Mendelian genetics, inheritance of traits, and so on. Hence it was easy for the public to see the "cracking" of the genetic code as an unraveling of a computer program, and the discovery of the double-helix structure of the DNA molecule as an explication of a computer's basic wiring diagram. The coupling of such a conceptual framework to one that sees man as a physical object virtually compels the conclusion that man may be designed and engineered to specification.

There is no point in complaining that Einstein never intended his theory to serve as one half of the metaphor just described. It is, after all, necessary for the two contexts coupled by a metaphor to be initially disjoint, just as (as I insisted earlier) a model must not have a causal connection to what it models. The trouble with the two metaphoric usages we have cited is that, in both, the metaphors are overextended. Einstein meant to say that there is no fixed, absolute spacetime frame within which physical events play out their destinies. Hence every description of a physical event (and, in that sense, of anything) must be relative to some specified spacetime frame. To jump from that to "everything is relative" is to play too much with words. Einstein's contribution was to demonstrate that, contrary to what had until then been believed, motion is not absolute. When one deduces from Einstein's theory that, say, wealth and poverty are relative, in that it is not the absolute magnitudes of the incomes of the rich and poor that matters, but the ratios of one to the other, one has illicitly elevated a metaphor to the status of a scientific deduction.

The example from molecular biology illustrates an overextension of a metaphor in another sense; there the extent of what we know about the human as a biological organism is vastly exaggerated. The result is, to say the least, a premature closure of ideas. The metaphor, in other words, suggests the belief that everything that needs to be known is known.

The computer has become a source of truly powerful and often useful metaphors. Curiously, just as with the examples already cited, the public embrace of the computer metaphor rests on only the vaguest understanding of a difficult and complex scientific concept (here, the theory of computability and the results of Turing and Church concerning the universality of certain computing schemes). The public vaguely understands—but is nonetheless firmly convinced—that any effective procedure can, in principle, be carried out by a computer. Since man, nature, and even society carry out procedures that are surely "effective" in one way or another, it follows that a computer can at least imitate man, nature, and society in all their procedural aspects. Hence everything (that word again!) is at least potentially understandable in terms of computer models and metaphors. Indeed, on the basis of this unwarranted generalization of the words "effective" and "procedure," the word "understanding" is also redefined. To those fully in the grip of the computer metaphor, to understand *X* is to be able to write a computer program that realizes *X*. This is vividly exemplified by Professor Marvin Minsky, Director of M.I.T.'s Artificial Intelligence Laboratory, who writes,

> [For computers] "to write really good music or draw highly meaningful pictures will of course require better *semantic* models in these *areas*. That these are not available is not so much a reflection on the state of heuristic [computer] programs as on the traditionally disgraceful state of analytic criticism in the arts—a cultural consequence of the fact that most esthetic analysts wax indignant when it is suggested that it might be possible to understand what they are trying to understand."[2]

Clearly, what Minsky means by "understanding" music or painting is quite different from what, say, Mozart or Picasso meant by the same term. One of his uses of the word "understand" in this quoted passage is essentially a pun—though, I belive, an unconscious one—

on the other. The very innocence of his use of it testifies to the tenacity of the hold the metaphor has on him.*

This new definition of understanding is now very widely accepted, not only explicitly in scientific circles, but implicitly in the common wisdom. It implies, as the psychologist George A. Miller ruefully pointed out,

> "that the only reason something cannot be done by a universal Turing machine is that we don't understand it. Given this interpretation of what 'understanding' consists of, any attempt to suggest counterexamples becomes merely a confession of ignorance or, if one persists in claiming that he can understand something he cannot describe explicitly, one becomes a prototypical member of that class of people known as mystics."[3]

In other words, the computer metaphor has become another lamppost under whose light, and only under whose light, men seek answers to burning questions.

No branch of science has erected this lamppost more deliberately and with more enthusiasm than has psychology. George Miller writes:

> "Many psychologists have come to take for granted in recent years . . . that men and computers are merely two different species of a more abstract genus called 'information processing systems.' The concepts that describe abstract information processing systems must, perforce, describe any particular examples of such systems."[4]

The narrowing of vision that characterizes modern scientific investigation, just like the narrowing of the field of view accomplished by a microscope, can only be justified, and sustained, because it permits us to see things we could otherwise not see. Science and technology have, after all, momentous achievements to their credit. Given the depth to which the computer metaphor has pene-

* It must be said that Prof. Minsky did not adopt the computer metaphor because of a naive misunderstanding of the theory of computability and its implications. To the contrary, as a deservedly acknowledged authority in computer science, he adopted the metaphor thoughtfully and deliberately.

trated psychology, it is natural to ask whether it has justified itself there in terms of some tangible achievements.

The impact that the computer, in its role as a high-speed numerical calculator, has had on psychology, although undoubtedly very large, hardly counts, as a "tangible achievement." Psychology has long tried to become "scientific" by imitating that most spectacularly successful science, physics. Psychologists, however, seemed for a very long time to have misunderstood just what it was that made physics somehow more a science than psychology. Like sociology too, psychology mistook the most superficial property of physics, its apparent preoccupation with numbers and mathematical formulas, for the core that makes it a science. Large sections of psychology therefore tried to become as mathematical as possible, to count, to quantify, to identify its numbers with variables (preferably ones having subscripted Greek letters), and to manipulate its newfound variables in systems of equations (preferably differential equations) and in matrices just as the physicists do. The very profusion of energy expended on this program was bound to guarantee that some useful results would be achieved. Psychometrics, for example, is and remains an honorable trade. And there can be no question that statistics benefited enormously from the exercise it was given by psychology, just as it had benefited in the days of its infancy from its application to gambling. Perhaps it repaid its two patrons about equally.

It is often said that the computer is merely a tool. The function of the word "merely" in that statement is to invite the inference that the computer can't be very important in any fundamental sense because tools themselves are not very important. I have argued that tools shape man's imaginative reconstruction of reality and therefore instruct man about his own identity. Yet the folk wisdom that perceives the computer as a basically trivial instrument rests on an accurate insight: the computer, used as a "number-cruncher" (that is, merely as a fast numerical calculator, and it is so used especially in the behavioral sciences), has often, as George Miller has also pointed out, put muscles on analytic techniques that are more powerful than the ideas those techniques enable one to explore. "The methodological rigorists," writes Stanislav Andreski, "are like cooks who would

show us all their shiny stoves, mixers, liquidizers and what not, without ever making anything worth eating."[5] The high-speed number-cruncher is, in the hands of many psychologists, merely their newest, shiniest, and most spectacular mixer-liquidizer.

When the computer is used merely as a numerical tool in psychology (or in any other field), it does not usually create a focusing of vision; i.e., it is not comparable to the microscope. It is therefore unrealistic to expect such use to uncover previously unseen worlds or to render distinct what was earlier seen only in vague outline. Because the view of man as a species of the more general genus "information-processing system" does concentrate our attention on one aspect of man, it invites us to cast all his other aspects into the darkness beyond what that view itself illuminates. We are entitled to ask what we would purchase at that cost.

There can be no final answer to such a question, for the extent of the creative analogical reach of a metaphor must, by its very nature, be always surprising and thus not fathomable in advance. We can say in anticipation that the power of a metaphor to yield new insights, depends largely on the richness of the contextual frameworks it fuses, on their potential mutual resonance. How far that potential will be realized depends, of course, on how profoundly the participants in the creative metaphoric act can command both contexts. That is why, for example, the computer expert who knows nothing but computers (the "Fach Idiot," as the Germans call such a person) can derive no broad intellectual nourishment from his expertise and is therefore doomed to remain forever a hacker. That is also why the computer metaphor is, as George Miller puts it, "most productive in areas where a considerable foundation of theory based on previous research already exists."[6]

One area of psychology was extraordinarily well-prepared to benefit from a fusion with the kind of process-oriented thinking characteristic of computer scientists; it was the area which concerns itself with the cognitive processes underlying the acquisition and memorization of information. An enormous amount of laboratory work had been done on, for example, the task of memorizing so-called nonsense syllables. One form of an experiment that has been performed by countless psychological laboratories is to present a

subject with, say, a dozen pairs of three-letter syllables, one pair at a time, and to ask him, on each (but the first) presentation of the first of the pair, to say what the second is. The syllables are carefully chosen to be inherently meaningless. Thus, for example, CAT is not a nonsense syllable, but PAG is. Subjects are exposed to the list, one pair at a time, repeatedly until they are able to give the correct response item to each stimulus item. Edward S. Feigenbaum reported,

> "The phenomena of rote learning are well-studied, stable, and reproducible. For example, in the typical behavioral output of a subject, one finds:
>
> a. Failures to respond to a stimulus are more numerous than overt errors.
> b. Overt errors are generally attributable to confusion by the subject between similar stimuli or similar responses.
> c. Associations which are given correctly over a number of trials sometimes are then forgotten, only to reappear and later disappear again. This phenomenon has been called oscillation.
> d. If a list *x* of syllables or syllable pairs is learned to the criterion; then a list *y* is similarly learned; and finally retention of list *x* is tested; the subject's ability to give the correct *x* responses is degraded by the interpolated learning. The degradation is called retroactive inhibition. The overt errors made in the retest trial are generally intrusions from the list *y*. The phenomenon disappears rapidly. Usually after the first retest trial, list *x* has been relearned back to criterion.
> e. As one makes the stimulus syllables more and more similar, learning takes more trials."[7]

Feigenbaum, currently a professor of computer science at Stanford University, conjectured that this sort of learning task involved the subject in an active, complex symbol-manipulation process which could best be described and understood in terms of more elementary symbol-manipulation processes of just the kind that can be programmed for a computer.

Of course, nothing would have been easier than to write a small program for a computer which would have enabled an experi-

menter to give the computer lists of nonsense syllables that the computer could then reproduce perfectly after the first "trial." The task Feigenbaum set for himself was much harder: to produce, in the form of a computer program, a model of cognitive processes whose over-all behavior would closely approximate that of human subjects engaged in memorizing nonsense syllables, and whose detailed internal functions would constitute a theoretical explanation of the difficulties actually observed in experiments. Moreover, he wished his explanations to be at least consistent with such psychological observations as, for example, that humans have both short-term memories, in which they can apparently hold a few symbols for instant recall during a short period of time, and longer-term memories, in which an almost unlimited amount of information can be stored but from which individual items can be retrieved only at the expense of some effort. If this "effort" to remember is thought of as the computational effort involved in executing a perhaps long subroutine, it becomes obvious how one can begin to apply the computer metaphor.

Feigenbaum's central idea is for the computer to store *descriptions* of the syllables presented to it, not the actual syllables themselves. The syllable DAX, for example, may be described by the fact that its first letter has a vertical leading edge and contains a closed loop, that its second letter contains a horizontal middle bar, and so on. When a syllable is first presented to the system, a description of it just sufficiently detailed to allow it to be discriminated from the syllables already stored is added to storage. If it is a stimulus item, that is, the first of a syllable pair, then a "cue" consisting of a minimal description of the syllable with which it is to be associated is stored with it. Because all these descriptions are so minimal, the system often makes wrong associations when presented with stimulus items. But because the correct response item is presented whenever the system makes such an error, the descriptive information then in play may be improved by adding further description to it. Eventually the system learns the list perfectly. When another list is then attempted, the descriptions associated with it may again be confused with those corresponding to the first list, and vice versa. The system is thus capable of exhibiting retroactive inhibition. And

clearly, as the items to be learned are made more and more similar to one another, an increasing number of trials is required to refine the discriminating power of each relevant descriptor. The system thus behaves very much as does a human confronted with the same task.

Feigenbaum's program, though by now very old (it was completed in 1959), remains instructive in at least two respects. First, it offers us a relatively simple example of what is meant by a model of a cognitive process in computer-program form. The way it organizes its information storage is meant to be a functional description of the human intermediate-term memory. As such, it explains, for example, how it may be that we can totally forget something for a long time and yet recall it again under certain circumstances. It cannot be that the allegedly forgotten item was simply wiped out of our minds, for if it were, we could never regain access to it. In Feigenbaum's model no information is ever destroyed. But information may be hidden, so to speak, by descriptors leading to other associations; thus one memory may screen or mask another. Sometimes a refinement of the screening image (that is, of a cue) is, in Feigenbaum's system, all that is required to uncover (that is, to make again accessible) what was previously masked.

Feigenbaum's system also requires that the two syllables to be associated with one another be simultaneously available to the computer (that is, present in its store) for a short time. After a "cue" to the response item has been associated with the description of the stimulus syllable, the two syllables per se can be erased from the computer's store—in other words "forgotten." There is thus a part of his system that plausibly corresponds to what little psychologists know about the function of the human short-term memory. No one, least of all Feigenbaum, claims that his model constitutes "the" explanation for such phenomena. But it is an explanation in a domain where explanations are rare.

The second respect in which Feigenbaum's program is instructive is that it behaves in ways which were not directly and deliberately "programmed in," as the saying goes. For example, the program exhibits what psychologists call interference; that is, the acquisition of a new association interferes with the production of an

older one when the syllables involved have closely similar descriptions. The program contains no interference subroutine as such. The phenomenon arises as a consequence of the entire structure of the program, and appeared as a surprise to its designer. In that respect, then, the model *predicted* a behavioral phenomenon, which enormously enhanced its plausibility. The program thus instructs us that the easy and much-repeated slogan "a computer does only what its programmer told it to do" is in certain respects quite wrong and is in any case problematical.

The program we have been discussing is a member of a class of programs called "simulation programs." Their object is to simulate the way humans accomplish certain tasks, but decidedly not to accomplish those tasks in the most efficient way a computer possibly could. We have noted, for example, that a computer could easily be programmed to "memorize" lists of nonsense syllables in one "trial." But that would teach us nothing about how humans might accomplish what appears at least superficially to be the same task.

Because programs which concern the cognitive aspects of human behavior fall naturally within the domain of artificial intelligence, AI (about which more later), they need be distinguished from another class of AI programs, namely, ones that are entirely task-oriented.

Workers in AI tend to think of themselves as working in one of two modes, often called *performance mode* and *simulation mode*. Perhaps the best way to make the distinction clear is by analogy to flying. Virtually all early attempts to understand flying or to build flying models were based on imitating the flight of birds. It is a plausible conjecture that the myth of Icarus, the Greek hero who flew with wings attached to his body by wax and who crashed when the heat of the sun melted the wax, reflects man's early failure to imitate the birds. We might say that the early thinkers and pioneers were operating in simulation mode. Already by the middle of the nineteenth century, however, men like Henson and Stringfellow, and somewhat later Langley, shifted to what we might call performance mode. They considered that their task was to build flying machines based on whatever principles they could discover. Their aim

was performance first and understanding only to the extent that it would contribute to performance.

A third mode of operation should perhaps be mentioned in this context: theory mode. There were great aerodynamicists before there were practical aircraft. Lord Rayleigh, for example, published important papers specifically on the theory of flight beginning about 1875. Of course, after the Wright brothers achieved their historic flights in 1903, interest in aerodynamics increased continually and has not flagged to this day. But whereas the aerodynamicist is devoted to theory as such and tends to think of practical aircraft as being mere models of his theories, the aircraft designer looks to theory as being merely another source of ideas which may help him gain more performance from his machines.

The situation in AI is closely analogous to that just described. The goal of a majority of workers in AI is to build machines that behave intelligently, whether or not what they produce sheds any light on human intelligence. They are working in performance mode. They wish to build machines that speak as humans do and that understand human speech, that can, with the aid of television eyes and mechanical arms and hands, screw nuts onto bolts and assemble even more complex mechanical gadgets, that can analyze and synthesize chemical compounds, that can translate natural languages from one to another, that can compose music and complex computer programs, and so on. They are, of course, happy to accept whatever contributions the theoreticians (for example, the psychologists) can make that may help them realize their wishes. But their goal, unlike that of the theoretician who seeks understanding (or claims to), is performance first and last.

A program like Feigenbaum's clearly eschews performance; it is designed to require many trials to learn its lists, whereas, as I have pointed out, if performance were its goal, it could be made to memorize them in one trial.

The dividing line between simulation mode and performance mode is, as might be expected, not absolute. Often the only way to begin thinking about how to get a computer to do a specific task is to ask how people would do it. One thus starts out essentially simulating one's own introspectively observed behavior.

There is, of course, a difference between a program whose avowed aim is performance but that, at least initially, simulates "the way people do it," and a program that simulates what people do in order to learn something about people. But when a program undertaken under the latter banner succeeds, it also performs. Sometimes its authors then cannot resist the temptation to make performance as well as theoretical claims for it, and thus to contribute to the blurring of the line dividing performance mode from simulation mode. Newell, Shaw, and Simon, for example, wrote a program which could prove some theorems in the propositional calculus by simulating the way students who are naive about logic struggled with such proofs.[8] Newell, Shaw, and Simon stated as their aim, "we wish to understand how a mathematician, for example, is able to prove a theorem even though he does not know when he starts how, or if, he is going to succeed." After reporting how long it took their program to prove a number of theorems, they remarked: "One can invent 'automatic' procedures for producing proofs . . . but these turn out to require computing times of the order of thousands of years for the proof of [some particular theorem]." It is hard to read that statement as anything other than a claim that their program can perform usefully, aside from its being possibly instructive about how "mathematicians prove a theorem." As it happened, within a year or two after the appearance of their paper, the mathematician Hao Wang published an "automatic procedure," that is, a computer program, capable of proving all theorems in the propositional calculus. It proved the particular theorem whose proof Newell, Shaw, and Simon estimated would "require computing times of the order of thousands of years" in 1/4 second on what today would be considered a very primitive computer.[9]

The fuzziness of the line dividing simulation made from performance mode is, quite justifiably, a matter of little concern to workers in AI. At the outset of a large research effort, what is important is to have a fairly clear idea of at least the general domain within which questions are to be asked, or, to put it another way, of what it is that is not presently understood that the research is intended to help us understand. Wang's research yielded a result that deepened our understanding of certain aspects of mathematical logic. The aim of the work reported by Newell, Shaw, and Simon was, in their own words, to "understand the complex processes (heuristics) that are effective in problem solving."[10] They chose to examine how people prove theorems merely as an example of human problem solving. Newell and Simon have, as we shall see, pursued their work on problem solving to this very day. What has been problematical about it, and remains so, is in what sense of the word "understand" it helps us to understand man as an information processor or as anything else. That same question can usefully be asked about artificial intelligence generally.

That I have so far cited only very early AI projects is not in any sense "unfair," for AI researchers themselves continue to cite these very examples (namely, Feigenbaum's program, and the logic theory machine of Newell, Shaw, and Simon) as being fundamental work, as far as they go. Newell and Simon have, as I have said, continued their work on problem solving. Meanwhile other workers, notably those at M.I.T.'s and Stanford University's Artificial Intelligence Laboratories have increasingly turned their attention to robotics, that is, to the problems associated with the building of machines that sense aspects of their environments, e.g., with the aid of television eyes, and that are capable physically acting on it, e.g., by means of computer-controlled mechanical arms and hands. Their work has, as might be expected, generated a host of subproblems in such areas as vision, computer understanding of natural language, and pattern recognition.

Of course, some of these subproblems are also autonomous problems, that is, are independent of the research goals of robotics. Natural-language understanding by computer is an example of a problem that is inherently interesting and difficult in its own right. That any progress on it may prove useful for instructing robots is, to many workers, merely an additional motivation, certainly not the principal one. I shall later have more to say about the manipulation of natural language by computers. For the moment, however, let us turn to some of the more recent work on problem solving, particularly that of Newell and Simon.

The modern literature on problem solving is punctuated by two important books, George Polya's *How to Solve It* and Newell's

and Simon's *Human Problem Solving.*[11] Polya's book was first published in 1945, that is, years before electronic computers became practical research instruments. Yet in it Polya lays the groundwork and, in a sense, heralds all the work on problem solving that was to follow for thirty years afterward. Polya's concern is with heuristic problem-solving methods, that is, with those rules of thumb which, when applied, may well lead to a solution of the problem at hand or to some progress toward solving it, but which do not guarantee a solution. Heuristics are thus not algorithms, not effective procedures; they are plausible ways of attacking specific problems. Polya anticipated much of the later work of computer scientists on problem solving when he wrote:

> "Modern heuristic endeavors to understand the process of solving problems, especially the *mental operations typically useful in this process.* . . . Experience in solving problems and experience in watching other people solving problems must be the basis on which heuristic is built. In this study, we should not neglect any sort of problem, and should find out common features in the way of handling all sorts of problems; we should aim at general features, independent of the subject matter of the problem. The study of heuristic has 'practical' aims; a better understanding of the mental operations typically useful in solving problems. . . .
>
> "It is emphasized that all sorts of problems, especially PRACTICAL PROBLEMS, and even PUZZLES, are within the scope of heuristic [sic]. It is also emphasized that infallible RULES OF DISCOVERY are beyond the scope of serious research. Heuristic discusses human behavior in the face of problems. . . . Heuristic aims at generality, at the study of procedures which are independent of the subject-matter and apply to all sorts of problems."[12]

No clearer charter for the work of Newell and Simon could have been written. Polya, in effect, predicted those aspects of what Newell and Simon were later to do which most truly characterize their conception: the endeavor to understand mental operations, the emphasis on generality, on independence from subject matter, and on the usefulness of watching people solve problems, and the stress laid on puzzle-solving behavior. Finally, Polya emphasizes that his

book is about methods, and that the most important heuristic is "the end suggests the means."

What Newell and Simon were later to call "the means-ends method" was first suggested when the way an early version of their logic-theory machine proved theorems was compared with recordings of "thinking aloud" sessions of nonmathematics students attempting the same tasks. These so-called *protocols* proved highly suggestive for further work. Protocol taking, that is, watching other people solve problems, became virtually a hallmark of Newell and Simon's procedure.

The new information-processing psychology proceeds from the basic view

> "that programmed computer and human problem solver are both species belonging to the genus 'Information Processing System' (IPS). . . .
>
> "When we seek to explain the behavior of human problem solvers (or computers for that matter), we discover that their flexibility—their programmability—is the key to understanding them. Their viability depends upon their being able to behave adaptively in a wide range of environments. . . .
>
> "If we carefully factor out the influences of the task environments from the influences of the underlying hardware components and organization, we reveal the true simplicity of the adaptive system. For, as we have seen, we need postulate only a very simple information processing system in order to account for human problem solving in such tasks as chess, logic, and cryptarithmetic. The apparently complex behavior of the information processing system in a given environment is produced by the interaction of the demands of that environment with a few basic parameters of system, particularly characteristics of its memories.
>
> "Matters are simple, not because the law of large numbers cancels things out, but because things line up in a means-ends chain in which only the end points count (i.e., equifinality)."[13]

This is a truly remarkable statement, especially in light of Simon's claims that the hypothesis it represents "holds even for the whole man." It behooves us to attempt to understand just what this

"very simple" information-processing system is which produces complex behavior as a function of its environment and "a few basic parameters." We must also ask what it is about tasks like chess, logic, and cryptarithmetic that generalizes to the "whole range [of problems] to which the human mind has been applied," that is, to that range to which these same authors have promised computers will be applied "in the visible future."[14] The last question is especially pertinent because existing heuristic problem-solving programs deal only with very simple problems in chess and logic. Cryptarithmetic hardly counts here, since it is what is called, even in AI circles, a "toy problem."*

We have already agreed that it is entirely proper and even useful to assume a very particular viewpoint and, from the perspective it affords, to see man as an information processor. And since the computer, the Turing machine, is a universal information processor, it is natural to compare man as seen from that perspective with the computer. Information-processing *psychology* is, however, *not* information-processing *neurophysiology*. It does not attempt explanations in terms of bits or by making analogies to flip-flops, electronic circuits, and so on. It eschews, and rightly so, even explanations that depend on comparison with the sort of symbol manipulations that classical Turing machines do, e.g., writing, reading, erasing, and comparing extremely simple and irreducible symbols such as zeros and ones.

Recall that a program for a particular computer is essentially a description of another computer, that it transforms the former machine into the latter. One can therefore design a computer, and subsequently implement it in the form of a computer program, whose "built-in" elementary information processes (*eip's*—the terminology is Newell and Simon's) are ones that operate on arbitrarily

* True, there are very powerful programs for doing extremely complex symbolic logic and mathematics. But these are special-purpose programs which—although they may have benefited from AI techniques in their early development—can in no way be seen as the kinds of information-processing system Newell and Simon talk about. They are, beyond dispute, no more relevant to psychology than are the many programs which solve systems of differential equations. The currently most powerful chess programs were also written in performance mode, and, although they may use certain of the techniques of the AI armamentarium, they too are essentially irrelevant to psychology.

complex symbol structures, that is, that read, write, erase, compare such symbol structures with one another, and so on. Such structures can be made to represent formulas in logic, mathematical expressions, words, sentences, architectural drawings, etc., and, of course, computer programs which may themselves then be manipulated by eip's.

A particularly useful programming device is, for example, to organize information in the form of concatenations of individual items. The "link" that chains one item to the next is a machine address, a pointer, that is stored next to one of the items and points to its successor. A list (as such chains are commonly called) of items so concatenated may then again be considered an item and may thus be pointed to by still another item. In this way, structures of very great complexity may be created and manipulated. Specific eip's may treat them as single items, whereas others may course over them, inserting and deleting substructures, for example.

An information-processing system is therefore, in this context, a hardware computing system together with a program capable of executing eip's on stored symbol structures. It has, of course, input-output equipment, such as console typewriters, that enable adequate communication with the world outside itself.

The most ambitious information-processing system that has been built for the purpose of studying human problem-solving behavior is Newell and Simon's General Problem Solver (GPS).[15]

> "The main methods of GPS jointly embody the heuristic of means-ends analysis. . . . Means-ends analysis is typified by the following kind of common-sense argument:
>
> I want to take my son to nursery school. What's the difference between what I have and what I want? One of distance. What changes distance? My automobile. My automobile won't work. What is needed to make it work? A new battery. What has new batteries? An auto repair shop. I want the repair shop to put in a new battery; but the shop doesn't know I need one. What is the difficulty? One of communication. What allows communication? A telephone . . . and so on.

This kind of analysis—classifying things in terms of the functions they serve, and oscillating among ends, functions required, and means to perform them—forms the basic system of heuristic of GPS. More precisely, this means-ends system of heuristic assumes the following:

1. If an object is given that is not the desired one, differences will be detectable between the available object and the desired object.
2. Operators affect some features of their operands and leave others unchanged. Hence operators can be characterized by the changes they produce and can be used to try to eliminate differences between the objects to which they are applied and desired objects.
3. If a desired operator is not applicable, it may be profitable to modify its inputs so that it becomes applicable.
4. Some differences will prove more difficult to affect than others. It is profitable, therefore, to try to eliminate 'difficult' differences, even at the cost of introducing new differences of lesser difficulty. This process can be repeated as long as progress is being made toward eliminating the more difficult differences."[16]

To see how this works on one of the kinds of problems to which GPS has actually been applied, consider the following cryptarithmetic puzzle:

$$\begin{array}{r} DO \\ +\ IT \\ \hline TTD \end{array}$$

A subject is told that the above is an encoding of a problem in ordinary addition. Each letter represents a number, and no two letters represent the same number. His task is to assign numbers to the

letters in such a way that the given expression represents a correct addition. He is to produce a protocol, that is, to say out loud what he is thinking. Following is one possible such protocol, interspersed with an analysis in GPS-like terms:

Subject: $D + I$ must be greater than 9 because there is a carry to the next column.

Analysis: The subject applied the operator "process column."

Subject: T must be 1 since it is a carry.

Analysis: The subject applied the operator "assign value." He has reached a subgoal and reduced the difference between the given and the desired object. The "given object" is now

$$\begin{array}{r} DO \\ +\ I\ 1 \\ \hline 11\ D \end{array}$$

Subject: O must be at least 2.

Analysis: The subject applied the operator "generate possible values" to O. (There must have been some unspoken tentative application of the operator "assign value" whose results were rejected.)

Subject: Let's try $O = 2$.

Analysis: The subject applied the operator "assign value." Another reduction of difference. The "given object" is now

$$\begin{array}{r} 32 \\ +\ I\ 1 \\ \hline 113 \end{array}$$

Subject: $I = 8$.

Analysis: The "assign value" operator is applied and the difference between the given object and the desired object removed. The goal is reached.

This is a much simpler problem than those typically given to subjects and to GPS. A much more typical example of a problem that has been fully analyzed is

$$\begin{array}{r} DONALD \\ +\,GERALD \\ \hline ROBERT, \end{array}$$

where $D = 5$. The example we have worked out suffers from the additional fault that it does not display any wrong moves, backtracking, and so on. Nevertheless it gives a general, if pale, idea of the way GPS works and of what a protocol is.

It should also be understood that GPS is not the model of Newell and Simon's theory. GPS implies more about a distinct level of generality independent of the tasks to be accomplished than their theory requires. Indeed, there does not exist any one computer program that is a model of their theory. Instead there exists a number of programs, by no means all of them composed by Newell and Simon or their co-workers, that are substantially consistent with the theory and that employ the "main methods of GPS" listed above. It is the information-processing theory of man which concerns us here, not GPS as such. And we are concerned with that theory precisely because it, in one variation or another, sometimes explicitly and sometimes implicitly, underlies almost all the new information-processing psychology and constitutes virtually a dogma for the artificial-intelligence community.

The basic conclusions the theory reaches are the following.

> "All humans are information processing systems, hence have certain basic organizational features in common; all humans have in common a few universal structural characteristics, such as nearly identical memory parameters. These commonalities produce common characteristics of behavior among all human problem solvers.
>
> "Since the information processing system [i.e., the human seen as an information-processing system—J. W.] can be factored into (1) basic structure, and (2) the contents of long-term memory [i.e., programs and data], it follows that any proposal for commonality among problem solvers not attributable directly to basic structure must be represented as an identity or similarity in the contents of the long-term memories—in the production system or in other stored memory structures."
>
> [The theory] "proposes a system that, given enough time, can absorb any specification whatsoever—can become responsive to the full detail, say, of an encyclopedia (or a library of them). Hence the theory places the determination of differences and similarities of behavior directly upon the causes defining the content that will be stored in the human long-term memory. But these determinants of content are largely contingent upon the detail of the individual's life history. This does not mean that the determining processes are arbitrary or capricious or unlawful. It means that the contents can be as varied as the range of physical, biological, and social phenomena that surround the individual and from which he extracts them."[17] *

What is so remarkable about these conclusions is their scope, e.g., that the system the theory proposes, presumably a GPS-like system, can absorb any specification whatsoever. This claim—and what else can it reasonably be called?—is consistent with others of the authors' claims, namely, that the theory can account for the whole man and that computers will, within the visible future, handle problems over the whole range of human thought. The absurdity of what is being claimed for a GPS-like system is underscored by Newell and Simon's assertion that "The apparently complex behavior of the information processing system in a given environment is produced by the interaction of the demands of that environment with a few basic parameters of the system, particularly characteristics of its memories."[18] This is of course entirely consistent with their belief that man (like the ant) is "quite simple." But in this context a

* These statements invite comparison with B. F. Skinner's: "A scientific analysis of behavior must, I believe, assume that a person's behavior is controlled by his genetic and environmental histories rather than by the person himself as an initiating, creative agent" (from his *About Behaviorism*, New York, Alfred A. Knopf, New York, 1974, p. 189). The only difference between Skinner's position and that of the theory under discussion—and this difference is important from one point of view but totally irrelevant from another—is that Skinner refuses to look inside the black box that is the person, whereas the theory sees the inside as a computer.

technical claim is being made, namely, that GPS is quite simple, in the sense that, by changing a *few* of its parameters, its interaction with its environment will produce appropriately varied behavior simulating that of man.

In ordinary technical discussion we speak of a system being sensitive to "a few parameters" when the whole of its relevant mode of behavior can be entirely predetermined by setting a few switches or by entering a few data into its information store. A ship's navigation computer is of this type, for example. It will navigate the ship anywhere given only the geographical coordinates of its destination, some weather data, and so on. But to convert a GPS system from a chess player, say, to a cryptarithmetic puzzle solver is not a matter of changing a few numbers. In effect, the entire "memory structure" of GPS has to be replaced whenever GPS is to switch from one task to another. In other words, GPS is essentially nothing more than a programming language in which it is possible to write programs for certain highly specialized tasks. But, unless a computer program is to be considered a single parameter, GPS does not constitute any support for the claim that the complexity of human behavior is a function of only the human environment and a few parameters internal to the human information-processing system.

Occasionally, Newell and Simon do express a note of caution as when, for example, they admit that "we do not know what part of all human problem-solving activity employs a problem space, but over the range of tasks and individuals we have studied—a broad enough spectrum to make the commonalities nontrivial—a problem space is always used." But then, such mild disclaimers are countered by statements such as, "In spite of the restricted scope of the explicit evidential base of the theory, we will put it forth as a general theory of problem solving, without attempting to assess the boundaries of its applicability," and "we believe that the theory we are putting forth is much broader than the specific data on which we are erecting it."[19]

It is precisely this unwarranted claim to universality that demotes their use of the computer, computing systems, programs, etc., from the status of a scientific theory to that of a metaphor. They themselves say it: "Something ceases to be metaphor when detailed

calculations can be made from it; it remains metaphor when it is rich in features in its own right, whose relevance to the object of comparison is problematic."[20] The question then is, can detailed calculations be made from their "theory"? (I shall continue to use the word "theory" here, since it would be too awkward to always write "alleged theory" when referring to the work in question.)

The answer seems, at first glance, to be a resounding "yes." Is not Newell and Simon's book filled with examples of calculations made by GPS? But there is a subtle point here, a point of great importance, a point almost universally overlooked by workers in artificial intelligence who also believe themselves to be in possession of genuine theories. This point is perhaps most clearly illuminated by contrasting Feigenbaum's rote-memory simulator with the GPS programs reported in Newell and Simon's book. Feigenbaum's program is, as I said earlier, a model of a psychological theory, that is, of how people struggle with the task of memorizing nonsense syllables. The program itself is also a theory, as I pointed out; for example, if it is given to a psychologist who is familiar with the programming language in which it is written, one may expect that he will understand it. The property it has which qualifies it as theory, however, is that it enunciates certain principles from which consequences may be drawn. These principles themselves are in computer-program form, and their consequences emerge in the behavior of the program, that is, in the computer's reading of the program. Among them are the well-known phenomena of interference and retroactive inhibition that I mentioned earlier.

The situation is entirely different when, say, the logic-theory program is run in GPS. To be sure, the LOGIC THEORIST is again a theory (albeit a quite trivial one), specifically, a theory of how novices go about solving certain elementary logic problems. But GPS, and this is the crucial point, *is merely a framework within which the logic-theory program runs.* GPS is, in effect, a programming language which it is relatively easy to write logic-theory programs, cryptarithmetic programs, and so on. The elementary information processes, the eip's, which constitute its elementary instructions are simply the primitive instructions of the machine into which GPS has transformed its host computer. GPS as such does

not contain any principles—unless one counts as principles such observations as that, to solve problems, one must operate in terms of very general symbolic structures representing objects, operators, features of objects, and differences between objects, that one must build up a library of methods, and so on. Even then, GPS does not permit one to draw consequences from such "principles."

To say that GPS is, in any sense at all, an embodiment of a theory of human problem solving is equivalent to saying that high-school algebra is also such an embodiment. It too is a language, a computing schema, within which one can represent a theory already arrived at by other means. There is, of course, a theory of algebra. And there are theories of programming languages. But neither pretends to say anything about the psychology of human problem solving.

The counterargument to the above thesis is that the theory proposes a system—a GPS-like system—that embodies "a commonality among problem solvers" in its basic structure. It is that basic structure, that embodiment of the commonality among problem solvers, which makes it relatively easy to write problem-solving programs in a variety of quite disparate areas, e.g., logic and cryptarithmatic, in a GPS-like system. But, as Newell and Simon themselves said, any such commonality, if not attributable directly to basic structure, must be represented either in the program written in the GPS formalism or in the stored memory structures. In fact, all current versions of GPS-like systems have such absolutely minimal structure that the information that must be given them (in the form of program and data) for any particular problem-solving task must be detailed and specific, i.e., must define what the relevant operators are, to what objects they may be applied, what "difference" they make when applied to the proper object, and so on. As Newell and Simon say

> "Due account must be taken of the limitations of GPS's access to the external world. The initial part of the explicit instructions to GPS have been acquired long ago by the human in building up his general vocabulary. This [information] has to be spelled out to GPS."[21]

There, precisely, is where the question is begged. For the real question is, what happens to the whole man as he builds his general vocabulary? How is his perception of what a "problem" is shaped by the experiences that are an integral part of his acquisition of his vocabulary? How do those experiences shape his perception of what "objects," "operators," "differences," "goals," etc., are relevant to any problem he may be facing? And so on. No theory that sidesteps such questions can possibly be a theory of human problem solving.

The dream of the artificial intelligentsia—a happy phrase the world owes to Dr. Louis Fein—is, of course, to bring into the world "machines that think, that learn, and that create," and whose ability to do these things will increase until "the range of problems they can handle will be coextensive with the range to which the human mind has been applied," as Drs. Newell and Simon already announced in 1958.[22] Their book was published fourteen years later and, as they promised, "the [machines'] ability to do these things increased rapidly," although the then "visible future" appears not to have arrived yet. But the vision is still clear enough. Now, indeed, they have told us how the trick is to be achieved. The proposed system, given enough time (but within the visible future), will become responsive to the full detail of a library of encyclopedias. In order for it to become thus responsive, however, it too will have to acquire a general vocabulary comparable to that commanded by an adult human; it will have to master natural language and internalize a fund of knowledge coextensive with that commanded by the human mind. A large segment of the artificial-intelligence community is, in fact, concentrating on the problem of computer understanding of natural language. That is the problem I intend to discuss in the next chapter.

For the moment, however, it remains to ask what image of man as problem solver can engender—I will not say justify—the mind-boggling vision here presented? To answer that question, we must look, first of all, at what Newell and Simon mean by a "problem."

Newell and Simon write,

> "If we provide a representation for [what is desired, under what conditions, by means of what tools and operations, starting with

> what initial information, and with what access to resources], and assume that the interpretation of [the symbol structures that represent this information] is implicit in the program of the problem-solving information-processing system, then we have defined a problem."[23]

And then, of course, since "all humans are information-processing systems," one can apply to them and to their affairs the "main methods of GPS," that is, "heuristic means-ends analysis," the testing of objects to see if they are "undesired" and therefore yet to be transformed by operators into the "desired objects," and so on.

It may be objected that such a characterization of the aims of artificial intelligence is a playing with words that unjustly overstates AI's actual and much more modest goals, that Newell and Simon and the AI community generally are really only talking about a certain class of technical problems to which the above definition applies and for which GPS-like methods are surely appropriate. But the point is precisely that the pervasion—we might well say perversion—of everyday thought by the computer metaphor has turned every problem into a technical problem to which the methods here discussed are thought to be appropriate. I shall have more to say on that theme later.

Let it suffice for now to note that H. A. Simon had already written in 1960,

> "Let us suppose that a specific technological development permits the automation of psychiatry itself, so that one psychiatrist can do the work formerly done by ten. . . . This example will seem entirely fanciful only to persons not aware of some of the research now going on into the possible automation of psychiatric processes."[24]

The research he had in mind was that then just begun by Kenneth Mark Colby, a psychoanalyst, who wrote,

> "Having conducted many laboratory experiments on free-association and having had years of clinical experience with neurotic processes, my initial hope was to simulate both [!] the free-associative thought characteristic of a neurotic process and its changes under the influence of a psychotherapist's interventions."[25]

The project—happily—failed. But Simon's words were to ring in Dr. Colby's ears for another six years before emerging again from his own pen. As we have already noted, my own work on the ELIZA system rekindled his enthusiasm and moved him to write the passages I quoted earlier but which bear repetition here:

> "If the [ELIZA] method proves beneficial, then it would provide a therapeutic tool which can be made widely available to mental hospitals and psychiatric centers suffering a shortage of therapists. . . . several hundred patients an hour could be handled by a computer system."[26]

Just as Simon predicted, and then some! Of course, this euphoric promise is predicated precisely on a view of man as a GPS-like machine. As Dr. Colby said,

> "A human therapist can be viewed as an information processor and decision maker with a set of decision rules which are closely linked to short-range and long-range goals. . . . He is guided in these decisions by rough empiric rules telling him what is appropriate to say and not to say in certain contexts."[27]

The patient is, in other words, an object different from the desired object. The therapist's task is to detect the difference, using difference-detecting operators, and then to reduce it, using difference-reducing operators, and so on. That is his "problem"! And that is how far the computer metaphor has brought some of us.

Notes to Chapter 1

1. Alexander Marschak, *The Roots of Civilization* (New York: Macmillan, 1972), p. 57.
2. L. Mumford, *Technics and Civilization* (New York: Harcourt Brace Jovanovich, 1963), p. 14.
3. *Ibid*, pp. 13, 14.
4. *Ibid.*, p. 15.
5. Marschak, *op. cit.*, p. 14.
6. C. Pearson, *The Grammar of Science* (London: Dent, 1911), p. 11.
7. J. W. Forrester, in M. Greenberger, ed., *Managerial Decision Making in Management and the Computer of the Future* (Cambridge, Mass.: M.I.T. Press, 1962), pp. 52-53.

Notes to Chapter 2

1. Alan M. Turing, "On Computable Numbers, with an Application to the *Entscheidungsproblem*," *Proc. London Math. Soc. Ser.* 2-42 (Nov. 17, 1936), pp. 230-265.
2. M. Minsky, *Computation, Finite and Infinite Machines* (Englewood Cliffs, N.J.: Prentice-Hall, 1967), p. 111.
3. M. Polanyi, *The Tacit Dimension* (New York: Doubleday, 1966), p. 4.

Note to Chapter 3

1. From J. Weizenbaum, "Contextual Understanding by Computers," *Communications of the ACM*, vol. 10, no. 8 (August 1967), pp. 474-480.

Notes to Chapter 4

1. Fyodor Dostoevski, *The Gambler*, quoted by E. Bergler, *The Psychology of Gambling* (New York: Hill and Wang, 1957), p. 33.
2. Bergler, *op. cit.*, p. 9.
3. *Ibid.*, p. 230.
4. *Ibid.*, p. 230.
5. M. Polanyi, *Personal Knowledge* (New York: Harper Torchbooks, 1964), p. 291.
6. *Ibid.*, p. 292.
7. Aldous Huxley, *Science, Liberty, and Peace* (New York: Harper, 1946), pp. 35-36.
8. H. A. Simon, *The Sciences of the Artificial* (Cambridge, Mass.: The M.I.T. Press, 1969), pp. 24-25.
9. Huxley, *op. cit.*, pp. 36-37.
10. Simon, *op. cit.*, pp. 52-53.

Notes to Chapter 5

1. For a brief and readily understandable discussion of Chomsky's position, see his *Problems of Knowledge and Freedom* (New York: Pantheon Books 1971), especially Chapter I. More complete and considerably more technical discussions are to be found in his *Aspects of the Structure of Syntax* (Cambridge, Mass.: The M.I.T. Press, 1965), and in the references there cited.

2. H. A. Simon and A. Newell, "Heuristic Problem Solving: The Next Advance in Operations Research," *Operations Research*, vol. 6 (Jan.-Feb. 1958), p. 8.
3. A. Kaplan, *The Conduct of Inquiry* (San Francisco, Calif.: Chandler, 1964), p. 296. This important book contains an excellent discussion of models and theories in the social sciences. See especially chapters VII and VIII.
4. K. B. Krauskopf and A. Besier, *Fundamentals in Physical Science* (New York: McGraw-Hill, 6th ed., 1971), p. 28.
5. Kaplan, *op. cit.*, p. 57.
6. A. Newell and H. A. Simon, *Human Problem Solving* (Englewood Cliffs, N.J.: Prentice-Hall, 1972), p. 10.
7. J. von Neuman, *The Computer and the Brain* (New Haven, Conn.: Yale University Press, 1958), p. 82.
8. P. Suppes, "Meaning and Uses of Models," in B. H. Kazemier and D. Vuysje, eds., *The Concept and the Role of the Model in Mathematics and Natural and Social Sciences* (New York: Gordon and Breach, 1961), p. 172.

Notes to Chapter 6

1. I. A. Richards, *The Philosophy of Rhetoric* (Oxford, England: Oxford University Press, 1936), p. 93.
2. Marvin Minsky, ed., *Semantic Information Processing* (Cambridge, Mass.: The M.I.T. Press, 1968), p. 12. From the introduction by M. Minsky.
3. G. A. Miller, *Language, Learning, and Models of the Mind.* Unpublished manuscript. June 1972.
4. *Ibid.*
5. S. Andreski, *Social Science as Sorcery* (New York: St. Martin's Press, 1972), p. 114.
6. Miller, *op. cit.*
7. Edward A. Feigenbaum, "The Simulation of Verbal Learning Behavior," in E. A. Feigenbaum and J. Feldman, eds., *Computers and Thought* (New York: McGraw-Hill, 1963), p. 299.
8. A. Newell, J. C. Shaw, and H. A. Simon, "Empirical Explorations of the Logic Theory Machine: A Case Study in Heuristics" (The **RAND** Corp., March 1957), Report P-951.
9. Hao Wang, "Toward Mechanical Mathematics," in K. M. Sayre and F. J. Coosson, eds., *The Modeling of Mind* (Notre Dame, Ind.: University of Notre Dame Press, 1963), pp. 91-120.
10. Newell *et al., op. cit.*
11. G. Polya, *How to Solve It* (Copyright 1945 by Princeton University Press; © 1957 by G. Polya; Princeton Paperback 1971). A. Newell and H. A. Simon, *Human Problem Solving* (Englewood Cliffs, N.J.: Prentice-Hall, 1972).
12. Polya, *op. cit.*, pp. 129-133 (emphases are Polya's).
13. Newell and Simon, *op. cit.*, pp. 870-871.
14. H. A. Simon and A. Newell, "Heuristic Problem Solving: The Next Advance in Operations Research," *Operations Research*, vol. 6 (Jan.-Feb. 1958), pp. 8ff.

15. This system is described in some detail in Newell and Simon, *op. cit.*, particularly in Chapter 9, "Logic: GPS and Human Behavior," pp. 455-554.
16. *Ibid.*, p. 416.
17. *Ibid.*, pp. 864-866.
18. *Ibid.*, p. 809.
19. *Ibid.*, pp. 791, 809.
20. *Ibid.*, p. 5.
21. *Ibid.*, p. 855.
22. Simon and Newell, *op. cit.*, p. 8.
23. Newell and Simon, *Human Problem Solving*, p. 73.
24. H. A. Simon, "The Shape of Automation" (1960), reprinted in Z. W. Pylyshyn, ed., *Perspectives on the Computer Revolution* (Englewood Cliffs, N.J.: Prentice-Hall, 1970), p. 413.
25. K. M. Colby, "Simulations of Belief Systems," Chapter 6 in R. C. Schank and K. M. Colby, eds., *Computer Models of Thought and Language* (San Francisco, Calif.: W. H. Freeman and Co., 1973), p. 257.
26. K. M. Colby, J. B. Watt, and J. P. Gilbert, "A Computer Method of Psychotherapy: Preliminary Communication," *The Journal of Nervous and Mental Disease*, vol. 142, no. 2 (1966), pp. 148-152.
27. *Ibid.*, p. 150.

Notes to Chapter 7

1. R. K. Lindsay, "Inferential Memory as the Basis of Machines which Understand Natural Language," in E. A. Feigenbaum and J. Feldman, eds., *Computers and Thought* (New York: McGraw-Hill, 1963), p. 218.
2. B. F. Green, A. K. Wolf, C. Chomsky, and K. Laughery, "Baseball, An Automatic Question Answering System," in Feigenbaum and Feldman, *op. cit.*, pp. 207-216.
3. D. G. Bobrow, "Natural Language Input for a Computer Problem-Solving System," in M. Minsky, ed., *Semantic Information Processing* (Cambridge, Mass.: The M.I.T. Press, 1968), pp. 135-215.
4. J. Weizenbaum, "Contextual Understanding by Computers," *Communications of the ACM*, vol. 10, no. 8 (Aug. 1967), pp. 474-480.
5. See, however, Chapter III, p. 106. There **ELIZA** has a script that permits it to understand in a much stronger sense.
6. Roger C. Schank, "Identifications of Conceptualizations Underlying Natural Language," in R. C. Schank and K. M. Colby, eds., *Computer Models of Thought and Language* (San Francisco, Calif.: W. H. Freeman and Co., 1973), p. 191.
7. J. Weizenbaum, *op. cit.*, p. 476.
8. Terry Winograd, "A Procedural Model of Language Understanding," in Schank and Colby, *op. cit.*, p. 154.
9. *Ibid.*, p. 155f and p. 163.
10. *Ibid.*, p. 185.
11. G. A. Miller, *Language, Learning, and Models of the Mind* (unpublished manuscript, June 1972).

12. R. C. Schank, "Conceptual Dependency: A Theory of Natural Language Understanding," *Cognitive Psychology*, no. 3 (1972), pp. 553-554.
13. *Ibid.*, pp. 553, 629.

Notes to Chapter 8

1. In M. Greenberger, ed., *Management and the Computer of the Future* (Cambridge, Mass.: The M.I.T. Press, 1962), p. 123.
2. David C. McClelland, "Testing for Competence Rather Than for 'Intelligence,'" *American Psychologist*, vol. 28, no. 1 (January 1973), pp. 1-14.
3. From Greenberger, *op. cit.*, p. 118.
4. See especially the work of R. A. Spitz, "Hospitalism," in *Psychoanalytic Study of the Child*, vol. 1, 1945.
5. E. Erikson, *Childhood and Society* (New York: W. W. Norton, 2d ed., 1963), pp. 79, 80.
6. *Ibid.*, pp. 75-76.
7. For an account of the findings in this area, see Robert E. Ornstein, *The Psychology of Consciousness* (San Francisco, Calif.: W. H. Freeman and Co., 1972). Chapter III is particularly relevant to the present discussion. It is written in plain English. The references it cites open the door to the entire area of research.
8. Reprinted in *The World of Mathematics* (New York: Simon and Schuster, 1956), vol. IV, pp. 2041-2050. This important essay is very much worth a trip to the library, as is the set of volumes in which it appears.
9. J. Bruner, *On Knowing* (New York: Atheneum, 1973), pp. 3-5.
10. H. Wang, *From Mathematics to Philosophy* (New York: Humanities Press, 1974), p. 324. Kurt Gödel himself referred to this in December 1951 as one of "the two most interesting rigorously proved results about minds and machines." The other is that either there exist certain formal questions which neither humans nor machines can answer, or the human mind can answer some formal questions that machines cannot.
11. D. C. Denett, "The Abilities of Men and Machines." Paper delivered to the American Philosophical Association, December 29, 1970.
12. W. Caudill and H. Weinstein, "Maternal Care and Infant Behavior in Japan and in America," *Psychiatry* 32 (1967): 12-43. Reprinted in C. S. Lavatelli and F. Stendler, eds., *Readings in Child Behavior and Development* (New York: Harcourt Brace Jovanovich, 3d ed., 1972), p. 78.
13. *Ibid.*, pp. 80 *et seq.*
14. Diaz v. Gonzales, 261 U.S. 102 (1923), Per Holmes, O. W. I owe this reference to Professor Paul Freund of the Law School of Harvard University.

Notes to Chapter 9

1. On **DENDRAL**, see B. Buchanan, G. Sutherland, and E. A. Feigenbaum, "Heuristic **DENDRAL**: A Program for Generating Explanatory Hypotheses in Or-