

# Why People Think Computers Can't

Marvin Minsky

*MIT*

*Cambridge, Massachusetts*

MOST PEOPLE ARE CONVINCED computers cannot think. That is, *really* think. Everyone knows that computers already do many things that no person could do without “thinking.” But when computers do such things, most people suspect that there is only an illusion of thoughtful behavior, and that the machine

- doesn't know what it's doing
- is only doing what its programmer told it to
- has no feelings. And so on.

The people who built the first computers were engineers concerned with huge numerical computations: that's why the things were *called* computers. So, when computers first appeared, their designers regarded them as nothing but machines for doing mindless calculations.

Yet even then a fringe of people envisioned what's now called “Artificial Intelligence”—or “AI” for short—because they realized that computers could manipulate not only numbers but also *symbols*. That meant that computers should be able to go beyond arithmetic, perhaps to imitate the information processes that happen inside minds. In the early 1950's, Turing began a Chess program, Oettinger wrote a learning program, Kirsch and Selfridge wrote vision programs, all using the machines that were designed just for arithmetic.

Today, surrounded by so many automatic machines, industrial robots, and the R2-D2's of Star Wars movies, most people think AI is much more advanced than it is. But still, many “computer experts” don't believe that machines will

ever “really think.” I think those specialists are too used to explaining that there's nothing inside computers but little electric currents. This leads them to believe that there can't be room left for anything else—like minds, or selves. And there are many other reasons why so many experts still maintain that machines can never be creative, intuitive, or emotional, and will never really think, believe, or understand anything. This essay explains why they are wrong

## Can Computers Do Only What They're Told?

We naturally admire our Einsteins and Beethovens, and wonder if computers ever could create such wondrous theories or symphonies. Most people think that “creativity” requires some mysterious “gift” that simply cannot be explained. If so, then no computer can create—since, clearly, anything machines can do can be explained.

To see what's wrong with that, we'd better turn aside from those outstanding works our culture views as very best of all. Otherwise we'll fall into a silly trap. For, until we first have some good ideas of how we do the *ordinary* things—how ordinary people write ordinary symphonies—we simply can't expect to understand how great composers write great symphonies! And obviously, until we have some good ideas about *that*, we'd simply have no way to guess how difficult might be the problems in composing those most outstanding works—and then, with no idea at all of how they're made,

of course they'll seem mysterious! (As Arthur Clarke has said, *any* technology sufficiently advanced seems like magic.) So first we'd better understand how people and computers might do the ordinary things that we all do. (Besides, those skeptics should be made to realize that their arguments imply that ordinary people can't think, either.) So let's ask if we can make computers that can use ordinary common sense; until we get a grip on that we hardly can expect to ask good questions about works of genius.

In a practical sense, computers already do much more than their programmers tell them to. I'll grant that the earliest and simplest programs were little more than simple lists and loops of commands like "*Do this. Do that. Do this and that and this again until that happens*" That made it hard to imagine how more could emerge from such programs than their programmers envisioned. But there's a big difference between "impossible" and "hard to imagine." The first is about *it*; the second is about *you!*

Most people still write programs in languages like BASIC and FORTRAN, which make you write in that style—let's call it "do now" programming. This forces you to imagine all the details of how your program will move from one state to another, from one moment to the next. And once you're used to thinking that way, it is hard to see how a program could do anything its programmer didn't think of—because it is so hard to make that kind of program do *anything* very interesting. Hard, not impossible.

Then AI researchers developed new kinds of programming. For example, the "General Problem Solver" system of Newell, Shaw and Simon lets you describe processes in terms of statements like "if you're on the wrong side of a door, go through it"—or, more technically, "if the difference between what you have and what you want is of kind D, then try to change that difference by using method M."<sup>1</sup> Let's call this "do whenever" programming. Such programs automatically apply each rule whenever it's applicable—so the programmer doesn't have to anticipate when that might happen. When you write in this style, you still have to say what should happen in each "state" the process gets into—but you don't have to know in advance when each state will occur.

You also could do such things with the early programming language COMIT, developed by Yngve at MIT, and the SNOBOL language that followed it. Today, that programming style is called "production systems."<sup>2</sup> The mathematical theory of such languages is explained in my book.<sup>3</sup>

That "General Problem Solver" program of Newell and Simon was also a landmark in research on Artificial Intelligence, because it showed how to write a program to solve a problem that the programmer doesn't know how to solve. The trick is to tell the program what kinds of things to

TRY; you need not know which one actually will work. Even earlier, in 1956, Newell, Shaw, and Simon developed a computer program that was good at finding proofs of theorems in mathematical logic—problems that college students found quite hard—and it even found some proofs that were rather novel (It also showed that computers could do "logical reasoning"—but this was no surprise, and since then we've found even more powerful ways to make machines do such things.) Later, I'll discuss how this relates to the problem of making programs that can do "common-sense reasoning."

Now, you might reply, "Well, everyone knows that if you try enough different things at random, of course, eventually, you can do anything. But if it takes a million billion trillion years, like those monkeys hitting random typewriter keys, that's not intelligence at all. That's just Evolution or something."

That's quite correct—except that the "GPS" system had a real difference - it didn't do things randomly. To use it, you also had to add another kind of knowledge—"advice" about when one problem-state is likely to be better than another. Then, instead of wandering around at random, the program can seek the better states; it sort of feels around, the way you'd climb a hill, in the dark, by always moving up the slope. This makes its "search" seem not random at all, but rather purposeful. The trouble—and it's very serious—is that it can get stuck on a little peak, and never make it to the real summit of the mountain.

Since then, much AI research has been aimed at finding more "global" ways to solve problems, to get around that problem of getting stuck on little peaks which are better than all the nearby spots, but worse than places that can't be reached without descending in between. We've discovered a variety of ways to do this, by making programs take larger views, plan further ahead, reformulate problems, use analogies, and so forth. No one has discovered a "completely general" way to always find the very highest peak. Well, that's too bad—but it doesn't mean there's any difference here between men and machines—since people, too, are almost always stuck on local peaks of every kind. That's life.

Today, most AI researchers use languages like LISP, that let a programmer use "general recursion." Such languages are even more expressive than "do whenever" languages, because their programmers don't have to foresee clearly either the kinds of states that might occur or when they will occur; the program just constrains how states and structures will relate to one another. We could call these "constraint languages"<sup>4</sup>

Even with such powerful tools, we're still just beginning to make programs that can learn and can reason by analogy. We're just starting to make systems that will learn to recognize which old experiences in memory are most analogous to present problems. I like to think of this as "do something

---

<sup>1</sup>Of course, I'm greatly simplifying that history

<sup>2</sup>Allen Newell and Herbert Simon, *Human Problem Solving*

<sup>3</sup>Marvin Minsky, *Computation: Finite and Infinite Machines*, Prentice-Hall 1967

---

<sup>4</sup>This isn't quite true. LISP doesn't really have those "do whenevers" built into it, but programmers can learn to make such extensions, and most AI workers feel that the extra flexibility outweighs the inconvenience

sensible” programming. Such a program would remember a lot about its past so that, for each new problem, it would search for methods like the ones that worked best on similar problems in the past. When speaking about programs that have *that* much self-direction, it makes no sense at all to say “computers do only what they’re told to do,” because now the programmer knows so little of what situations the machine may encounter in its future—or what it will remember from its past.

A generation later, we should be experimenting on programs that *write better programs to replace themselves*. Then at last it will be clear how foolish was our first idea—that never, by their nature, could machines create new things. This essay tries to explain why so many people have guessed so wrongly about such things.

### Could Computers Be Creative?

I plan to answer “no” by showing that there’s no such thing as “creativity” in the first place. I don’t believe there’s any substantial difference between ordinary thought and creative thought. Then why do we think there’s a difference? I’ll argue that this is really not a matter of what’s in the mind of the artist—but of what’s in the mind of the critic; the less one understands an artist’s mind the more creative seems the work the artist does.

I don’t blame anyone for not being able to do the things creative people do. I don’t blame them for not being able to explain it, either. (I don’t even blame them for thinking that if creativity can’t be explained, it can’t be mechanized; in fact I agree with that.) But I do blame them for thinking that, just because they can’t explain it themselves, then no one *ever* could imagine how creativity works. After all, if you can’t understand or imagine how something might be done at all, you *certainly shouldn’t expect to be able to imagine how a machine could do it!*

What is the origin of all those skeptical beliefs? I’ll argue first that we’re unduly intimidated by admiration of our Beethovens and Einsteins. Consider first how hard we find it to express the ways we get our new ideas—not just “creative” ones but everyday ideas. The trouble is, when focussing on creativity, we’re prone to notice it when others get ideas that we don’t. But when we get our own ideas, we take them for granted, and don’t ask where we “get” them from. Actually we know as little—maybe less—of how we think of ordinary things. We’re simply so accustomed to the marvels of everyday thought that we never wonder—until unusual performances attract attention. (Of course, our superstitions about creativity serve other needs, e.g., to give our heroes special qualities that justify the things we ordinary losers cannot do.)

Should we suppose that outstanding minds are any different from ordinary minds at all, except in matters of degree? I’ll argue both ways. I’ll first say “No, there’s nothing special in a genius, but just some rare, unlikely combination of virtues—none very special by itself.” Then, I’ll say “Yes,

but in order to *acquire* such a combination, you need at least a lucky accident—and maybe something else—to make you able, in the first place, to acquire those other skills.”

I don’t see any mystery about that mysterious combination itself. There must be an intense concern with some domain. There must be great proficiency in that domain (albeit not in any articulate, academic sense). And one must have enough self-confidence, immunity to peer pressure, to break the grip of standard paradigms. Without that one might solve problems just as hard—but in domains that wouldn’t be called “creative” by one’s peers. But none of those seems to demand a basic qualitative difference. As I see it, any ordinary person who can understand an ordinary conversation must have already in his head most of the mental power that our greatest thinkers have. In other words, I claim that “ordinary, common sense” already includes the things it takes—when better balanced and more fiercely motivated—to make a genius. Then what makes those first-raters so much better at their work? Perhaps two kinds of difference-in-degree from ordinary minds. One is the way such people *learn* so many more and deeper skills.

The other is the way they learn to *manage* using what they learn. Perhaps beneath the surface of their surer mastery, creative people also have some special administrative skills that better knit their surface skills together. A good composer, for example, has to master many skills of phrase and theme—but those abilities are shared, to some degree, by everyone who *talks* coherently. An artist also has to master larger forms of form—but such skills, too, are shared by everyone who knows good ways to “tell a tale.” A lot of people learn a lot of different skills—but few combine them well enough to reach that frontal rank. One minor artist masters fine detail but not the larger forms; another has the forms but lacks technique.<sup>5</sup>

We still don’t know why those “creative masters” learn so much so well. The simplest hypothesis is that they’ve come across some better way to choose how and what to learn! What might the secret be? The simplest explanation: such a “gift” is just some “higher-order” kind of expertise—of knowing how to gain and use one’s other skills. What might it take to learn *that*? Obvious: *one must learn to be better at learning!*

If that’s not obvious, perhaps our culture doesn’t teach how to think about learning. We tend to think of learning as something that just happens to us, like a sponge getting soaked. But learning really is a growing mass of skills; we start with some but have to learn the rest. Most people never get deeply concerned with acquiring increasingly more advanced learning skills. Why not? Because they don’t pay off right away! When a child tries to spoon sand into a pail,

---

<sup>5</sup>Of course each culture sets a threshold to award to just a few that rank of “first class creativity”—however great or small the differences among contestants. This must make social sense, providing smallish clubs of ideal-setting idols, but shouldn’t then burden our philosophy with talk of “inexplicability.” There must be better ways to deal with feelings of regret at being “second-rate.”

the child is mostly concerned with filling pails and things like that. Suppose, though, by some accident, a child got interested in how that pail-filling activity itself improved over time, and how the mind's inner dispositions affected that improvement. If only once a child became involved (even unconsciously) in how to learn better, then that could lead to exponential learning growth.

Each better way to learn to learn would lead to better ways to build more skills—until that little difference had magnified itself into an awesome, qualitative change. In this view, first-rank “creativity” could be just the consequence of childhood accidents in which a person's learning gets to be a little more “self-applied” than usual. <sup>6</sup> If this image is correct, then we might see creativity happen in machines, once we begin to travel down the road of making machines that learn—and learn to learn better.

Then why is genius so rare? Well, first of all, the question might be inessential, because the “tail” of every distribution must be small by definition. But in the case of self-directed human thought-improvement, it may well be that all of us are already “close to some edge” of safety in some socio-biological sense. Perhaps it's really relatively easy for certain genes to change our brains to make them focus even more on learning better ways to learn. But quantity is not the same as quality—and, possibly, no culture could survive in which each different person finds some wildly different, better way to think! It might be true, and rather sad, if there were genes for genius that weren't hard at all for Evolution to come upon—but needed (instead of nurturing) a frequent, thorough weeding out, to help us keep our balance on some larger social scale.

### Can Computers Choose Their Own Problems?

Some people even ask “How could computers make mistakes?” as though, somehow, ability to err itself might be some precious gift. There's nothing wrong with seeking for some precious quality, but only some form of quiet desperation would lead one to seek for it in error and mistake. It seems to stem from the misconception that creativity is rooted in some chance or random element that can't be found in any well-defined machine. This is silly, first because machines can simulate random behavior as well as one can want, and, second because it doesn't explain the consistency and coherency with which creative people produce.

Another often-heard speculation: “I can see how a machine could solve very difficult problems that are given to it by someone. But isn't the very hardest and most important problem, really, to figure out *what problem to solve?* Perhaps the thing machines can't do is to invent their own problems?” This is wonderfully profound and silly at the same

---

<sup>6</sup>Notice, that there's no way a parent could notice—and then reward—a young child's reflective concern with learning. If anything, the kid would seem to be doing *less* rather than more—and might be urged to “snap out of it”

time. Really, it's usually *much* easier to think of good problems than to solve them—though sometimes it is profoundly hard to find exactly the right question to ask. In any case, a culture frames its history of ideas so that the rewards are largest for opening new areas. But the problems *inside* each subject can be just as *hard*.

The reason this speculation is wrong is that, in order to solve any really hard problem (by definition of “hard”), one has to find a way to break it down into other problems that one can solve. Therefore, the ability to invent and formulate new problems must already be a part of being reasonably intelligent. It only obscures the point to argue that those are “only sub-problems.” The ability to compose good questions is a requisite of intelligence, not a special *sine qua non* for creativity.

Besides, some people, more than others, prefer to look outside a present context and ask larger questions like “Am I working on the right problem?” But everyone *can* do this to some degree—and can be worse off by doing it excessively. I see nothing especially mysterious about that inclination to “take a larger view.”<sup>7</sup> The interesting problem is less in what generates the originality, and more in how we build control mechanisms that appropriately exploit and suppress it.

The rest of this essay explains the weaknesses of several other common theories of how machines must differ fundamentally from minds. Those theories are unproved today—not because of anything about machines, but just because we know too little about how human minds really work. We're simply not prepared to search for things that we can do but machines cannot. Because of this, we'll focus on a more constructive kind of question: why *people* are so very bad at making theories of what *they* can or cannot do!

### Can Computers Think Only Logically?

Our culture is addicted to images of minds divided into two parts. Usually, one mind-half is seen as calculating, logical, and pretty brittle; the other half seems sort of soft and vague. There are so many variants of this, and all so ill-defined, that it's impossible to tell them apart: let's call them Soft-Hard Dumbbell theories:

Logic	—	Intuition
Spatial	—	Verbal
Quantitative	—	Qualitative
Local	—	Global
Reason	—	Emotion
Thinking	—	Feeling
Literal	—	Metaphorical, etc.

There's nothing wrong with starting with two-part theories—if you use them as steps toward better theories. But

---

<sup>7</sup>That is, given the advanced abilities to plan, generalize, and make abstractions that all ordinary people possess; computers haven't exhibited much ability in these areas, yet

when you *stop* at dumbbell theories then, most likely, you have only one idea instead of two:

Whatever-it-is- -Everything else.

The trouble with one-part theories is that they don't lead anywhere, because they can't support enough detail. Most of our culture's mental-pair distinctions are stuck just so, which handicaps our efforts to make theories of the mind. I'm especially annoyed with recent fads that see minds as divided evenly into two halves that live within the left and right-hand sides of the brain:

LEFT-LIKE—RIGHT-LIKE  
(Computer-like)—(Rest-of-mind-like)

This is really neat. It not only supports beliefs that minds do things computers can't, but even provides a handy physical brain-location in which to put the differences!

Each half of the brain has dozens, and probably hundreds of different sections of machinery. There definitely are some differences between right and left. But these structural differences between corresponding parts of the right and left halves appear very much less than the differences *within* each half. Despite that flood of half-baked brain-half stories, I've heard of little evidence for systematic differences in how those left-right portions really function, in spite of all those newsstand magazines and books, and it would seem that even brain-scientists' theories about minds are just as naive as yours and mine. They're just as prone to "observe" whatever distinctions they imagine. Just for fun, I'll contribute two of my own speculations on what our brain-halves do:

MASTER-SLAVE THEORY: The two brain-sides at first develop more or less the same ways, in parallel. As time goes on and specialties mature, the need for order and coherency requires one to become dominant: a program cannot smoothly serve two masters. Whichever side acquires control, perhaps according to some inborn bias, the other side remains more "childish" and subservient, and used for cruder, simpler parts of whatever computation is involved.

DIFFERENCE THEORY: Our AI theories of thinking emphasize mechanisms for recognizing differences. This requires access to closely related pairs of descriptions. One must describe the present situation as it is; the other describes the ideal or goal—that is, the situation as one wishes it to be. What better way to do that than to slave together a pair of similar machines with, say, one side depicting more of what's perceived, the other representing anticipated or imagined goals.

Either image seems to suit those popular but vague descriptions of the two "dissected personalities," right and left, that emerge when surgeons split a patient's brain in halves. The right side, say, would be better at realistic, concrete things—things as they are. The left half would have specialized in long-range plans, in things that aren't yet, in short, at things we like to call "abstract."

Those age-old distinctions between Logic and Intuition, or Reason and Emotion, have been the source of many unsound arguments about machine intelligence. It was clear in AI's earliest days that logical deduction would be easy to program. Accordingly, people who imagined thinking to be mostly logical expected computers soon to do the things that people used their logic for. In that view, it ought to be much harder, perhaps impossible, to program more qualitative traits like intuition, metaphor, aesthetics or reasoning by analogy. I never liked such arguments.

In 1964, my student T.G. Evans finished a program to show that computers could actually use analogies. It did some interesting kinds of reasoning about perception of geometric structures. This made some humanistic skeptics so angry that they wrote papers about it. Some threw out the baby with the bath by seeming to argue that if machines could indeed do that kind of analogical reasoning, then, maybe that kind of reasoning can't be so important. One of them complained that Evans' program was too complicated to be the basis of an interesting psychological theory, because it used about 60,000 computer instruction-words (That seemed like saying there wasn't any baby in the first place.)

In any case Evans' program certainly showed it was wrong to assume computers could do only logical or quantitative reasoning. Why did so many people make that mistake? I see it as a funny irony: those critics had mistaken *their own personal limitations* for limitations of computers! They had projected their own inability to explain how either person or machine could reason by analogy onto the outer world, to suppose that *no* well-defined mechanism could do such a thing. In effect, they were saying that since *they* could see no explanation then, surely, there could *be* no explanation!

Another misconception stems from confusing different senses of logic. Too many computer specialists talk as though computers are perfectly logical, and that's all. What they really mean is that *they* can understand, using logic, how all those tiny little computer circuits work. But, just because the little circuits can be understood *by* logic doesn't mean *at all* that those circuits can only *do* logic! That's like thinking you could figure out what houses are *for* from knowing how bricks work.

Many AI workers have continued to pursue the use of logic to solve problems. This hasn't worked very well, in my opinion; logical reasoning is more appropriate for displaying or confirming the *results* of thinking than for the thinking itself. That is, I suspect we use it less for solving problems than we use it for explaining the solutions to other people and—much more important—to ourselves. When working with the actual details of problems, it is usually too hard to package the knowledge we need into suitably logical form. So then we have to use other methods, anyway—methods more suitable for the "networks of meanings" that I'll discuss shortly. Still, I consider such ideas to be of great importance in making theories of how we *represent* the things we

think about, and especially in how we think when we reason *carefully*.

### Could a Computer Really Understand Anything?

"I see you've programmed that computer to obey verbal commands. You've probably inserted into its memory how it should respond to each command. But I don't believe the program really *understands* the words, in any human sense."

This criticism is deserved by most computer systems around these days. But how does it apply to the 1965 program written by Daniel Bobrow that solves high-school algebra "word problems"? It could solve some problems like these:

The distance from New York to Los Angeles is 3000 miles. If the average speed of a jet plane is 600 miles per hour, find the time it takes to travel from New York to Los Angeles by jet

Bill's father's uncle is twice as old as Bill's father. Two years from now Bill's father will be three times as old as Bill. The sum of their ages is 92. Find Bill's age.

Most human students find problems like these quite hard. They find it easier to learn to solve the kinds of *equations* they encounter in high school algebra; that's just cook-book stuff. But to solve the word problems, you have to figure out what equations to solve. Doesn't this mean you have to understand at least something of what the words and sentences mean?

Well, to begin with, Bobrow's program used a lot of tricks. It guesses that the word "is" usually means "equals." It doesn't even try to figure out what "Bill's father's uncle" is, except to notice that this phrase resembles "Bill's father."<sup>8</sup> It doesn't know that "age" and "old" have anything to do with time, only that they're numbers to be put into, or found from, equations. Given these and a couple of hundred other facts about the words, it sometimes (and by no means, always) manages to get the answers right.

But dare one say that Bobrow's program really "understands" those sentences? If meaning isn't caught in several hundred different tricks—might not we still imprison it in several hundred thousand tricks? Is "understand" even an idea we can ask Science to deal with?

Here's how I like to deal with such questions. I feel no obligation to define such words as "mean" and "understand," just because others tried it for five thousand years! Our words are only *social things*; it's great when they combine to give us good ideas. But here, I think, they only point to a maze of unproductive superstitions, that only handicapped our predecessors when they tried to figure out what "meanings" are and how they get connected to our words. It is a wrong-headed enterprise, like asking people to agree on

---

<sup>8</sup>In fact, if there were one less equation, it would assume that they mean the same, because they're so similar.

what is "good," without considering each person's different hopes and fears.

Fortunately, as I will show, there isn't any need to try to capture "meanings" in such rigid, public ways. In fact, that would defeat our real purposes. This is because any psychologically realistic theory of meanings needs built-in ways to deal with individual differences between the people who are to do the "knowing."

### Could A Computer Know What Something Means?

We can't think very well about meaning without thinking about the meaning of something. So let's discuss what numbers mean. And we can't think about what numbers mean very well without thinking about what some particular number means. Take Five. Now, no one would claim that Bobrow's algebra program could be said to understand what numbers "really" are, or even what Five really is. It obviously knows something of arithmetic, in the sense that it can find sums like "5 plus 7 is 12." The question is—does it understand numbers in any *other* sense—say, what are 5 or 7 or 12—or, for that matter, what are "plus" or "is"? Well, what would *you* say if I asked, "What is Five"? I'll argue that the secret lies in that little word "*other*."

Early this century, the philosophers Russell and Whitehead suggested a new way to define a number. "Five," they said, is *the set of all possible sets with five members*. This set includes every set of Five ball-point pens, and every litter of Five kittens. The trouble was, this definition threatened also to include sets like "these Five words" and even "the Five things that you'd least expect." Sets like those led to so many curious inconsistencies and paradoxes that the theory had to be doctored so that these could not be expressed—and *that* made the theory, in its final form, too complicated for any practical use (except for formalizing mathematics, where it worked very well indeed). But, in my view, it offers little promise for capturing the meanings of everyday common sense. The trouble is with its basic goal: finding for each word some single rigid definition. That's fine for formalizing Mathematics. But for real life, it ignores a basic fact of mind: what something means to me depends to some extent on everything else I know—and no one else knows just those things in just those ways.

But, you might complain, when you give up the idea of having rigid, definitions, don't you get into hot water? Isn't ambiguity bad enough; what about the problems of "circular definitions," paradoxes, and inconsistencies? Relax! We shouldn't be *that* terrified of contradictions; let's face it, most of the things we people think we "know" are crocks already overflowing with contradictions; a little more won't kill us. The best we can do is just be reasonably careful—and make our machines careful, too—but still there are always chances of mistakes. That's life.

Another kind of thing we scientists tend to hate are circular dependencies. If every meaning depends on the mind it's in—that is, on all other meanings in that mind—then

there's no place to start. We fear that when some meanings form such a circle, then there would be no way to break into the circle, and everything would be too subjective to make good science.

I don't think that we should fear the fact that our meanings and definitions run around in vicious circles, each depending on the others. There's still a scientific way to deal with this: just start making new kinds of theories—about those circles themselves! You don't *have* to break into them—you only need to have good theories *about* them. Of course, this is hard to do, and likely to get complicated. It was to avoid complication that all those old theories tried to suppress the ways that meanings depend on one another. The trouble is, that lost all the power and the richness of our wondrous meaning-webs! Let's face another fact: our minds really *are* complicated, perhaps more so than any other structure Science ever contemplated. So we can't expect the old ideas to solve all the new problems.

Besides, speaking of breaking into the meaning-circle, many science-fiction writers have pointed out that no one ever really *wants* to get oneself inside another mind. No matter if that's the only hope of perfect communication—of being absolutely sure you understand exactly, at every level of nuance what other people mean. The only way you could do that is by becoming exactly like that person but even then the game is lost, since then you couldn't understand any more (perfectly, that is) just what it was that your old self had tried to say.

### What Is a Number, That a Mind Might Know It?

Now let's return to what numbers mean. This time, to make things easier, we'll think about Three. What could we mean by saying that Three hasn't any single, basic definition, but is a web of different processes that depend upon each other? Well, consider all the roles "Three" plays.

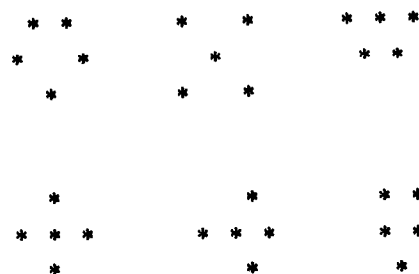
One way a person tells when there's a Three is to recite "One, Two, Three," while pointing to the different things. Of course, while doing that, you have to manage to (i) touch each thing once and (ii) not touch any twice. One easy way to do *that* is, to pick up one object, as you say each counting-word, and remove it. Soon, children learn to do that in their minds or, when it's too hard to keep track, to use some physical technique like finger-pointing.

Another way to tell a Three is to establish some Standard Set of Three things. Then you bring *your* set of things there and match them one-to-one: if all are matched and you have nothing left, then you had Three. And, again, that "standard Three" need not be physical; those three words, "One, Two, Three" would work quite well. To be sure, this might make it hard to tell which method you're using—"counting" or "matching"—at the moment. Good. It really doesn't matter, does it? (Except, perhaps, to philosophers.) For do-ers, it's really good to be able to shift and slip from one skill-process to another without even realizing it.

Another way to know a Three is by perceptual groups. One might think of Three in terms of arranging some objects into groups of One and Two. This, too, you can do mentally, without actually moving the objects, or you might lay them out on a table. You might learn several different such arrangements:



For Five you have more families of ways, because you can use groups of Two and Three, or groups of One and Four. A pentagon, a thing-filled square, a "W," a star, a plane, a cup; they all make Fives.



Another strand of understanding is to know how Three can be an incomplete or broken kind of Four—as in a defective square:



Which way is right—to count, or match, or group—which is the "real" meaning of a number? The very question shows how foolish is any such idea: each structure and its processes have both their own uses, and ways to support the others. This is what makes the whole into a powerful, versatile skill-system. Neither chicken nor egg need come first; they both evolve from something else.

It's too bad that so many scientists and philosophers despise such networks and only seek to construct simple "chains" of definitions in which each new thing depends only on other things that have been previously defined. That is what has given "reductionism" a bad name. The common sense meaning of Three is not a single link in one long chain of definitions in the mind. Instead, we simply let the word activate some rather messy web of different ways to deal with Threes of things, to use them, to remember them, to compare them, and so forth. The result of this is great for solving problems since, when you get stuck with one sense of meaning, there are many other things to try and do. If

your first idea about Three doesn't do some job, in some particular context, you can switch to another. But if you use the mathematician's way, then, when you get into the slightest trouble, you get completely stuck!

If this is so, then why *do* mathematicians prefer their single chains to our multiply-connected knowledge-nets? Why would anyone prefer each thing to depend upon as few other things as possible instead of as many as possible? The answer has a touch of irony: mathematicians *want* to get stuck! This is because, as mathematicians, we *want* to be sure above all that as soon as anything goes wrong, we'll be the first to notice it. And the best way to be sure of that is to make everything collapse at once! To mathematicians, that sort of fragility is *good*, not bad, because it helps us find out if any single thing that we believe is inconsistent with any of the others. This insures absolute consistency—and that is fine in Mathematics. It simply isn't good Psychology.

Perfect consistency is not so relevant to real life because—let's face it—minds will *always* have beliefs that turn out to be wrong. That's why our teachers use a very wrong theory of how to understand things, when they shape our *children's* mathematics, not into robust networks of ideas, but into those long, thin, fragile chains or shaky towers of professional mathematics. A chain breaks whenever there's just one single weak link, just as a slender tower falls whenever we disturb it just a little. And this could happen to a child's mind, in mathematics class, who only takes a moment to watch a pretty cloud go by.

The purposes of children, and of other ordinary people, are not the same as those of mathematicians and philosophers. They need to have as few connections as can be, to simplify their careful, accurate analyses. In real life the best ideas are those robust ones that connect to as many other ideas as possible. And so, there is a conflict when the teachers start to consult those academic technicians about curricula. If my theory's right, they're not just bad at that; they're just about as bad at that *as possible!* Perhaps this helps explain how our society arranges to make most children terrified of mathematics. We think we're making things easier for them to find what's right, by managing to make things go all wrong almost all the time! So when our children learn about numbers (or about anything else) I would prefer that they build meshy networks in their minds, not slender chains or flimsy towers. Let's leave that for when they take their graduate degrees.

For learning about Two, a pre-school child learns in terms of symmetry and congruence—two hands, two feet, two shoes—one doesn't need to count or refer to some standard ideal set. (It is only later that one learns that, every time you count, you get the same result.) We learn of Three in terms of rhymes and tales of Threes of Bears and Pigs and Turtle Doves (whatever those might be) that tell of many different *kinds* of Threes.

Note that those Bears are two and one, Parents and Child, while their famous bowls of porridge make a very different kind of Three—"too hot, too cold, just right"—

that shows the fundamental dialectic compromise of two extremes. (So do the bears' forbidden beds—too hard, too soft, just right.) Just think of all the different kinds of Threes that confront real children in the real world, and the complex network of how they all relate to one another in so many different, interesting ways. There simply isn't any sense to choosing one of them to be "defined" so as to come before the rest.

Our culture tries to teach us that a meaning really ought to have only a single, central sense. But if you programmed a machine that way, then, of course it couldn't really understand. Nor would a person either, since when something has just one meaning then it doesn't really "mean" at all because such mental structures are so fragile and so easy to get stuck that they haven't any real use. A network, though, yields gives many different ways to work each problem. And then, when one way doesn't work and another does, you can try to figure out why. In other words, the network lets you *think*, and thinking lets you build more network. For only when you have several meanings in a network is there much to think about; then you can turn things around in your mind and look at them from different perspectives. When you get stuck, you can try another view. But when a thing has just one meaning, and you get stuck, there's no way out except to ask Authority. That's why networks are better than logical definitions. There never is much meaning until you join together many partial meanings; and if you have only one, you haven't any.

### Could a Computer Know About the Real World?

Is there some paradox in this idea, that every meaning is built on other meanings, with no special place to start? If so, then isn't all a castle built on air? Well, yes and no. Contrary to common belief, *there's really nothing wrong at all with circular definitions.* Each part can give some meaning to the rest. There's nothing wrong with liking several different tunes, each one the more because it contrasts with the others. There's nothing wrong with ropes—or knots, or woven cloth—in which each strand helps hold the other strands together—or apart! There's nothing very wrong, in this strange sense, with having one's entire mind a castle in the air!

But then, how could such a mind have any contact with reality. Well, maybe this is something we must always face in any case, be we Machine or Man. In the human condition, our mental contact with the real world is really quite remote. The reason we don't notice this, and why it isn't even much of a practical problem, is that the sensory and motor mechanisms of the brain (that shape the contents of, at least, our infant minds) ensure enough developmental correspondence between the objects we perceive and those that lie out there in raw reality; and that's enough so that we hardly ever walk through walls or fall down stairs.

But in the final analysis, our idea of "reality" itself is rather network-y. Do triangles "exist" or are they only

Threes of Lines that share their vertices? What's real, anyway, about a Three—in view of all we've said; "reality" itself is also somewhat like a castle in the air. And don't forget how totally some minds, for better or usually for worse, *do* sometimes split away to build their own imaginary worlds. Finally, when we build intelligent machines we'll have a choice: either we can constrain them as we wish to match each and every concept to their outside-data instruments, or we can let them build their own inner networks and attain a solipsistic isolation totally beyond anything we humans could conceive.

To summarize: of course computers couldn't understand a real world—or even what a number is—were they confined to any single way of dealing with them. But neither then could child or philosopher. It's not a question of computers at all, but *only of our culture's foolish quest for meanings that can stand all by themselves, outside of any mental context.* The puzzle comes from limitations of the way our culture teaches us to think. It gives us such shallow and simplistic concepts of what it means to "understand" that—probably—no entity could understand *that way.* The intuition that our public has—that if computers worked that way, they couldn't understand—is probably quite right! But this only means we mustn't program our machines that way.

### Can a Computer Be Aware of Itself?

"Even if computers do things that amaze us, they're just mechanical. They can't believe or think, feel pain or pleasure, sorrow, joy. A computer can't be conscious, or self-aware—because it simply has no self to feel things with."

Well. What do you suppose happens in your head when someone says a thing like that to *you*? Do you understand it? I'll demonstrate that this problem, too, isn't actually about computers at all. It isn't even about "understanding." This problem is about you. That is, it turns around that little word "you." For when we feel that when we understand something, we also seem to think there must be some agent in our heads that "does" the understanding. When we believe something, there must be someone in our heads to do the believing. To feel, someone must do the feeling.

Now, something must be wrong with that idea. One can't get anywhere by assuming there's someone inside oneself—since then there'll have to be another someone inside that one, to do *its* understanding for it, and so on. You'll either end up like those sets of nested Ukrainian Russian dolls, or else you'll end up with some "final" inner self. In either case, as far as I can see, that leaves you just exactly where you started.<sup>9</sup> So what's the answer? The answer is—we must be asking the wrong question: perhaps we never had anything like "self-awareness" in the first place—but only thought we had it! So now we have to ask, instead—why do

we *think* we're self-aware?

My answer to this is that we are *not*, in fact, really self-aware. Our self-awareness is just illusion. I know that sounds ridiculous, so let me explain my argument very briefly. We build a network of half-true theories that gives us the illusion that we can see into our working minds. From those apparent visions, we think we learn what's really going on there. In other words, much of what we "discover" about ourselves, by these means, is just "made up." I don't mean to say, by the way, that those made-up ideas are *necessarily* better than or worse than theories we make about all other things that we don't understand very well. But I do mean to say that when we examine carefully the quality of the ideas most people have about their selves—ideas they got by using that alleged "self awareness"—we don't find that quality very good at all.

By the way, I'm not saying that we aren't aware of sounds and sights, or even of thoughts and ideas. I'm only saying that we aren't "self-aware." I'm also sure that the structures and processes that deserve to be called "self" and "awareness" are *very complicated concept-networks.* The trouble is that *those* are hardly at all like what we think they're like. The result is that in this area our networks don't fit together well enough to be useful for understanding our own psychology very well.

Now let's try to see what some of the meanings we attach to "self" are like. When you and I converse, it makes perfect sense for me to call you "you" and to call me "me." That's fine for ordinary social purposes, that is, when neither of us cares about the fine details of what is going on inside our minds. But everything goes wrong at once as soon as one's concerned with that—because those you's and me's conceal most of the intricacy of what's inside our minds that really do the work. The very purpose of such words like "you" and "self" is to symbolize away what we don't know about those complex and enormous webs of stuff inside our head.

When people talk, the physics is quite clear: I shake some air, which makes your ear-drums move, and some "computer" in your head converts vibrations into, say, little "phoneme" units. Next, oversimplifying, these go into strings of symbols representing words, so now somewhere in your head you have something that "represents" a sentence. The problem is, what happens next?

In the same way, when you see something, the waves of light excite your retinas, and this cause signals in your brain that correspond to texture fragments, bits of edges, color patches, or whatever. Then these, in turn, are put together (somehow) into a symbol-structure that "represents" a shape or outline, or whatever. What happens then?

We argued that it cannot help to have some inner self to hear or read the sentence, or little person, hiding there to watch that mental television screen, who then proceeds to understand what's going on. And yet that seems to be our culture's standard concept of the self. Call it the "Single

<sup>9</sup>Actually, there might be value in imagining the Self as like those dolls—each a smaller "model" of the previous system, and vanishing completely after a few stages.

Agent” theory: that inside every mind resides a certain special “self” that does the real mental work. Since this concept is so popular, we ought to have a theory of why we all believe such a ridiculous theory!

In fact, it isn’t hard to see why we hold onto such ideas—once we look past the single self and out into Society. For then we realize how valuable to us is this idea of Single Agent Self—no matter how simplistic, scientifically—in social matters of the greatest importance. It underlies, for instance, all the principles of all our moral systems; without it, we could have no canons of *responsibility*, no sense of blame or virtue, no sense of right or wrong. In short, without the idea of a Single Self, we’d scarcely have a culture to begin with. It also serves a crucial role in how we frame our plans and goals, and how we solve all larger problems—for, what *use* could solving problems be, without that idea of a self to savor and exploit their solutions.

And, furthermore, that image of a single self is central to the very ways we knit our personalities together—albeit though, as Freud has pointed out, it’s not the image of us as we *are* that counts, but as we’d like to *be*, that makes us grow. That’s why I didn’t mean to say that it is bad to have illusions for our Selves. (Why, what could one prefer to that, anyway?) And so, in short, no matter that it bollixes up our thinking about thinking; I doubt if we could survive without that wonderful idea of Single Self.

To build good theories of the mind, we’ll have to find a better way. We find that hard to do because the concept of the Single Self *is* so vitally important for those other reasons<sup>10</sup> But, just as Science forced us to accept the fact that what we think are single things—like rocks or mice or clouds—must sometimes be regarded as complicated other kinds of structures, we’ll simply have to understand that Self, too, is no “elementary particle,” but an extremely complicated construction.

We should be very used to this. There’s nothing wrong with the idea of Single Houses, either. They keep us warm and dry, we buy them and sell them, they burn down or blow away; they’re “things” all right but just up to a point. But when you really want to understand how Houses work, then you must understand that Houses aren’t really “things” at all but constructions. They’re made of beams and bricks and nails and stuff like that, and they’re also made of forces and vectors and stresses and strains. And in the end, you can hardly understand them at all without understanding the intentions and purposes that underlie the ways they’re designed.

So this wonderful but misleading Single Agent Self idea leads people to believe machines can’t understand, because it makes us think that understanding doesn’t need to be constructed or computed—only handed over to the Self—a thing that, you can plainly see, there isn’t room for in

---

<sup>10</sup>Similarly, we find Einstein’s space-time integration very difficult because, no matter how it bollixes up our thinking about Special Relativity, I doubt if we could survive without that wonderful idea of Separate Space

machines.

### Can a Computer Have a Self?

Now we can watch the problem change its character, before our eyes, the moment that we change our view. Usually, we say things like this:

A computer can’t do (xxx), because it has no self.

And such assertions often seem to make perfect sense—until we shed that Single Agent view. At once those sayings turn to foolishness, like this:

A computer can’t do (xxx), because all a computer can do is execute incredibly intricate processes, perhaps millions at a time, while constructing elaborately interactive structures on the basis of almost unimaginably ramified networks of interrelated fragments of knowledge

It doesn’t make so much sense any more, does it? Yet all we did was face one simple, complicated fact. The second version shows how some of our skepticism about computers emerges from our unwillingness to imagine what might happen in the computers of the future. The first version shows how some of our skepticism emerges from our disgracefully empty ideas about how *people* really work, or feel, or think.

Why are we so reluctant to admit this inadequacy? It clearly isn’t just the ordinary way we sometimes repress problems that we find discouraging. I think it is a deeper thing that makes us hold to that belief in precious self-awareness, albeit it’s too feeble to help us explain our thinking—intelligent or otherwise. It’s closer to a childish excuse—like “something made me do it,” or “I didn’t really mean to”—that only denies Single Self when fault or blame comes close. And rightly so, for questioning the Self is questioning the very notion of identity— and underneath I’m sure we’re all aware of how too much analysis could shred the fabrics of illusion that clothe our mental lives.

I think that’s partly why most people still reject computational theories of thinking, although they have no other worthy candidates. And that leads to denying minds to machines. For me, this has a special irony because it was only after trying to understand what computers— that is, complicated mechanisms—*could* do, that I began to have some glimpses of how a *mind* itself might work. Of course we’re nowhere near a sharp and complete theory of how human minds work—yet. But, when you think about it, how could we ever have expected, in the first place, to understand how minds work until after expertise with theories about very complicated machines? (Unless, of course, you had the strange but popular idea that minds aren’t complicated at all, only different from anything else, so there’s no use trying to understand them.)

I’ve mentioned what I think is wrong with popular ideas of self—but what ought we to substitute for that? Socially, as I’ve hinted, I don’t recommend substituting anything— it’s too risky. Technically, I have some ideas but this is not the

place for them. The “general idea” is to first develop better theories of how to understand the webs of processes we (or our machines) might use to represent our huge networks of fragments of common-sense knowledge. Once we’ve some of those that seem to work, we can begin work on other webs for representing knowledge about the first kind. Finally, we work on sub-webs –within those larger webs—that represent *simplified* theories of the entire mess! There’s no paradox at all in this, provided one doesn’t become too greedy—i.e., by asking that those simplified models be more than coarse approximations.

To do this will be quite complicated—but rightly so, for only such a splendid thing would seem quite worthy as a theory of a Self. For just as every child must connect a myriad of different ways to count and measure and compare, in order to understand that simple “concept of number”, so each child must surely build an even more intricate such network, in order that it understand itself (or even just a wishful image of itself) enough to grow a full-fledged personality. No less will do.

### Could a Computer Have Common Sense?

We all enjoy hearing those jokes about the stupid and literal behavior of computers, about how they send us checks for \$0.00 or bills for \$0.00 and so forth. Surely that total lack of common sense has encouraged most of us to doubt machines could have minds. It isn’t just that they do only what they’re told, it also that they’re so dumb that it’s almost impossible to tell them how to do things right.

And, indeed, those stories are quite true, on the whole. There certainly does seem something queer about computers. Why, for example, can they be so good at advanced mathematics, and stuff like that, so hard for us mortals—yet seem so dumb in general? You can hardly blame people for feeling that there must be some “vital missing element” in a computer!

On the surface, this seems to apply even to those AI programs. Isn’t it odd, when you think about it, that the very earliest AI programs excelled at “advanced, adult” subjects. I mentioned that the Newell-Simon program written in 1956 was quite good at certain kinds of Mathematical Logic. Then, in 1961, James Slagle wrote a program that could solve symbolic calculus problems at the level of college students (it got an A on an MIT exam). Around 1965 Bobrow’s program solved high-school algebra problems. And only around 1970 did we have robot programs, like Terry Winograd’s, which could deal with children’s building blocks well enough to stack them up, take them down, rearrange them, and put them in boxes.

Why were we able to make AI programs do such grown-up things so long before we could make them do childish things? The answer was a somewhat unexpected paradox. It seems that “expert” adult thinking is often<sup>11</sup> somehow simpler than children’s ordinary play! Apparently it can require more to be a novice than to be an expert, because

(sometimes, anyway) the things an expert needs to know can be quite few and simple, however difficult they may be to discover or learn in the first place. Thus, Galileo was very smart indeed, yet when he saw the need for calculus, he couldn’t manage to invent it. But any student can learn it today.

The knowledge network built into Slagle’s program had only some 100 “facts”—yet that’s enough to solve those college level problems. Most of these were simple facts about algebra and calculus, but some were about ways to tell *which of two problems is probably the easier*. Those were especially important because they embodied the program’s ability to make judgments about situations. Without them the program could only thrash about; with them it could usually make progress by making good decisions about what next to try.

Today we know a lot about making that sort of “expert” program, but we still don’t know nearly enough to build good common sense problem solving programs. Consider the kinds of things little children can do. Winograd’s program needed ways to combine different kinds of knowledge: about shapes and colors, space and time, words and syntax, and others, just to do simple things inside that “children’s world of building blocks”; in all it needed on the order of a thousand knowledge fragments, where Slagle needed only about a hundred—although the one just “played with toys” while the other could solve college level problems. As I see it, “experts” often can get by with deep but narrow bodies of knowledge—while common sense is almost always technically a lot more complicated.

Nor is it just a mere matter of quantity and quality of knowledge: Winograd needed more *different kinds* of ways for processes to control and exploit each other. It seems that common sense thinking needs a greater variety of different *kinds* of knowledge, and needs different *kinds* of processes. And then, once there are more different kinds of processes, there will be more different kinds of interactions between them, so we need yet more knowledge.

To make our robots have just their teeny bit of common sense, and that was nothing to write home about, our laboratory had to develop new kinds of programming—we called it “heterarchy,” as opposed to the “hierarchy” of older programs and theories. Less centralized, with more interaction and interruption between parts of the system, one part of Winograd’s program might try to parse a phrase while another part would try to rectify the grammar with the meaning. If one program guessed that “pick” is a verb, in “Pick up the block,” another program-part might check to see if “block” is really the kind of thing that *can* be picked up. Common sense requires a lot of that sort of switching from one viewpoint to another, engaging different kinds of ideas from one moment to another.

In order to get more common sense into our programs, I think we’ll have to make them more reflective. The present

---

<sup>11</sup>but certainly not always

systems seem to me a bit too active; they try too many things, with too little “thought.” When anything goes wrong, most present programs just back up to previous decisions and try something else—and that’s too crude a base for making more intelligent machines. A person tries, when anything goes wrong, to *understand* what’s going wrong, instead of just attempting something else. We look for causal explanations and excuses and—when we find them—add them to our networks of belief and understanding—we do intelligent learning. We’ll have to make our programs do more things like that.

### Can Computers Make Mistakes?

To err is human, etc. I’ll bet that when we try to make machines more sensible, we’ll find that *knowing what causes mistakes* is nearly as important as knowing what is correct. That is, in order to succeed, it helps to know the most likely ways to fail. Freud talked about censors in our minds, that serve to repress or suppress certain forbidden acts or thoughts; those censors were proposed to regulate much of our social activity. Similarly, I suspect that we accumulate censors for ordinary activities—not just for social taboos and repressions—and use them for ordinary problem solving, for knowing what *not* to do. We learn new ones, whenever anything goes wrong, by remembering some way to recognize those circumstances, in some “subconscious memory”—so, later, we won’t make the same mistake.<sup>12</sup>

Because a “censor” can only *suppress* behavior, their activity is invisible on the surface—except in making fewer blunders. Perhaps that’s why the idea of a repressive unconscious came so late in the history of psychology. But where Freud considered only emotional and social behavior, I’m proposing that they’re equally important in common-sense thinking. But this would also be just as hard to observe. And when a person makes some good intellectual decision, we tend to ask what “line of thought” lay behind it—but never think to ask “What thousand prohibitions warded off a thousand bad alternatives?”

This helps explain why we find it so hard to explain how our common sense thinking works. We can’t detect how our censors work to prevent mistakes, absurdities, bugs, and resemblances to other experiences. There are two reasons, in my theory, why we can’t detect them. First, I suspect that thousands of them work at the same time, and if you had to take account of them, you’d never get anything else done. Second, they have to do their work in a rather special, funny way, because they have to *prevent* a bad idea before you “get” that idea. Otherwise you’d think too slowly to get anywhere.

Accordingly, much of our thinking has to be unconscious. We can only sense—that is, have enough information to make theories about—what’s near the surface of our minds. I’m convinced that conscious thought is just one product

of complex “adversary processes” that go on elsewhere in the mind, where parts of thoughts are always under trial, with complicated presentations of the litigants, and lengthy deliberations of the juries.<sup>13</sup> And then, our “selves” hear just the final sentences of those unconscious judges.

How, after all, could it be otherwise? There’s no way any part of our mind could keep track of all that happens in the rest of our mind, least of all that “self”—that sketchy little model of the mind inside the mind. Our famous “selves” are valuable only to the extent they simplify and condense things. Each attempt to give “self consciousness” a much more comprehensive quality would be self defeating; like executives of giant corporations, they can’t be burdened with detail but only compact summaries transmitted from other agents that “know more and more about less and less.” Let’s look at this more carefully.

### Could a Computer Be Conscious?

When people ask that question, they seem always to want the answer to be “no.” Therefore, I’ll try to shock you by explaining why machines might be capable, in principle, of even more and better consciousness than people have.

Of course, there is the problem that we can’t agree on just what “conscious” means. Once I asked a student, “Can people be conscious?”

“Of course we can—because we are.”

Then, I asked: “Do you mean that you can know everything that happens in your mind?”

“I certainly didn’t mean *that* I meant something different.”

“Well,” I continued, “what did you mean by ‘conscious’ if you didn’t mean knowing what’s happening in your mind?”

“I didn’t mean conscious of what’s *in* my mind, just *of* my mind.”

Puzzled, I had to ask, “er, what do you mean?”

“Well, er, it’s too hard to explain.”

And so it goes. Why can we say so little about our alleged consciousness? Apparently because we can’t agree on what we’re talking about. So I’ll cheat and just go back to “self-awareness.” I’ve already suggested that although it is very useful and important, it really doesn’t do what we think it does. We assume we have a way to discover true facts about our minds but really, I claim, we only can make guesses about such matters. The arguments we see between psychologists show all too well that none of us have perfect windows that look out on mental truth.

If we’re so imperfect at self-explanation, then I don’t see any reason (in principle, at least) why we couldn’t make machines much better than we are ourselves at finding out about themselves. We could give them better ways to watch the ways their mechanisms work to serve their purposes and goals. The hardest part, of course, would lie not in acquiring such inner information, but in making the machine able

<sup>12</sup>More details of this theory are in my paper on *Jokes*

<sup>13</sup>Like the “skeptics” in Kornfeld’s thesis

to understand it—that is, in building programs with the common sense they'd need in order to be able to use such "insight" Today's programs are just too specialized, too dumb—if you'll pardon the expression—to handle anything as complicated as a theory of thinking. But once we learn to make machines smart enough to understand such theories, then (and only then) I see no special problem in giving them more "self-insight"<sup>14</sup>

Of course, that might not be so wise to do—but maybe we will have to. For I suspect our skeptics have things upside-down, who teach that self awareness is a strange, metaphysical appendage beyond and outside, mere intelligence, which somehow makes us human, yet hasn't any necessary use or function. Instead, it might turn out that, at some point, we *have to* make computers more self-conscious, just in order to make them smarter! It seems to me that no robot could safely undertake any very complex, long-range task, unless it had at least a little "insight" into its own dispositions and abilities. It ought not start a project without knowing enough about itself to be pretty sure that it will stay "interested" long enough to finish. Furthermore, if it is to be able to learn new ways to solve hard, new kinds of problems, it may need, again, at least a simplified idea of how it already solves easier, older problems. For this and other reasons, I suspect that any really robust problem solver, one that can adapt to major changes in its situation, must have some sort of model of itself.

On the other side, there are some minor theoretical limitations to the quality of self-insight. No interesting machine can, in general, predict ahead of time exactly what it will do, since it would have to compute faster than it can compute. So self-examination can yield only "general" descriptions, based on simplified principles. People, too, can tell us only fragments of details of how they think, and usually end up saying things like "It occurred to me." We often hear of "mystical experiences" and tales of total understanding of the self. But when we hear the things they say of what they learned— it seems they only learned to quench some question-asking portion of the mind.

So "consciousness" yields just a sketchy, simplified mind model, suitable only for practical and social uses, but not fine-grained enough for scientific work. Indeed, our models of ourselves seem so much weaker than they ought to be that one suspects that systematic mechanisms oppose (as Freud suggested) the making of too-realistic self-images. That could be to a purpose, for what would happen if you really could observe your "underlying" goals—and were to say

"well, I don't *like* those goals" and change them in some willy-nilly way? Why, then, you'd throw away an eon's worth of weeding out of non-survivors—since almost every new invention has some fatal bug. For, as we noted earlier, a part of Evolution's work is rationing the creativity of our

<sup>14</sup>I think that *we* are smart enough to understand the general principles of how we think, if they were told to us. Anyway, I sure hope so. But I tend to doubt that we have enough built-in, self-information channels to figure it out by "introspection."

brain-machines.

But when and if we chose to build more artfully intelligent machines, we'd have more options than there were in our own evolution—because biology must have constrained the wiring of our brains, while we can wire machines in almost any way we wish. So, in the end, those artificial creatures might have richer inner lives than people do. (Do I hear cries of "treason"?) Well, we'll just have to leave that up to future generations—who surely wouldn't want to build the things *that* well without good reasons to.

### Can We Really Build Intelligent Machines?

It will be a long time before we learn enough about common sense reasoning to make machines as smart as people are. We already know a lot about making useful, specialized, "expert" systems, but we don't yet know enough to make them able to improve themselves in interesting ways. Nevertheless, all those beliefs which set machine intelligence forever far beneath our own are only careless speculations, based on unsupported guesses on how human minds might work. The best uses for such arguments are to provide opportunities to see more ways that *human* minds can make mistakes! The more we know of why our minds do foolish things, the better we can figure out how we so often do things so well. In years to come, we'll learn new ways to make machines and minds both act more sensibly. We'll learn about more kinds of knowledge and processes, and how to make machines learn still more knowledge for themselves, while learning for ourselves to think of "thinking," "feeling" and "understanding" not as single, magic faculties, but as complex yet comprehensible webs of ways to represent and use ideas.

In turn, those new ideas will give us new ideas for new machines, and those, in turn, will further change our ideas on ideas. And though no one can tell where all of this may lead, one thing is certain, even now: there's something wrong with any claim to know, today, of differences of men and possible machines—because we simply do not know enough today, of either men or possible machines.

#### Automation of Reasoning: Classical Papers in Computational Logic (Vol. I and Vol. II)

Jörg H. Siekman, Graham Wrightson (eds.)

*It is reasonable to expect that the relationship between computation and mathematical logic will be as fruitful in the next century as that between analysis and physics in the last - John McCarthy, 1963*

Logic has emerged as one of the fundamental disciplines of computer science. Computational logic, which continues the tradition of logic in a new technological setting, has led to such diverse fields of application as automatic program verification, program synthesis, as well as logic programming and the fifth generation computer system.

This series of volumes, the first covering 1957 to 1966 and the second 1967 to 1970, contains those papers, which have shaped and influenced the field of computational logic and makes available the classical work. The main purpose of this series is to evaluate the ideas of the time and to select papers, which can be regarded as classics.

Contents: 60 original papers 3 survey papers Complete bibliography on computational logic To be published by Springer Verlag, Berlin, Heidelberg, New York, 1982