Invited Lecture

Prospects for Cognitive Science

Herbert A. Simon
Carnegie-Mellon University

PROFILE:
HERBERT A. SIMON is Richard King Mellon University Professor of Computer Science and Psychology at Carnegie-Mellon University, where he has taught since 1949.

Educated at the University of Chicago (B.A., 1936; Ph.D., 1943), he is a member of the National Academy of Sciences, and has received awards for distinguished research from the American Psychological Association. In 1978, he received the Alfred Nobel Memorial Prize in Economic Sciences, and in 1986 the National Medal of Science.

During the past thirty years he has been studying decision-making and problem-solving processes, using computers to simulate human thinking. He has published over 600 papers and 20 books and monographs.

The Program Committee, in inviting me to give this lecture, asked me to look ahead a little bit at the prospects over the next decade for cognitive science and artificial intelligence. It's somewhat easier to predict a decade in a science than it is to predict the next day on the stock market, or maybe even tomorrow's weather.

It took about twenty-six years, more than a quarter of a century, for physics to move from Planck's discovery of the black body law to the discovery of the basic laws of quantum mechanics. So, a ten-year look ahead doesn't seem to be very formidable, as long as I am not supposed to predict exactly what is going to come out of the research, but rather to talk about the directions which that research might take—what might be some of the exciting and profitable directions of research. And if we do that, we might be able to look ahead not only ten years, but a bit longer than that. We might even glimpse into the next century, which after all is only a dozen years away.

In my remarks, I am going to use the terms "cognitive science" and "artificial intelligence" more or less interchangeably. Both of those domains are concerned with producing intelligence—intelligent behaviour in computers. Cognitive science wishes to do so in order to gain a deeper understanding of our own human intelligence, while artificial intelligence is primarily interested in augmenting human intelligence, and therefore, is not concerned that the processes that are used in artificial intelligence programs should look very much like the processes that human beings use in their intelligent activities.

AI can use nano-second or pico-second speeds, but we know that speeds like that are unavailable to the human brain, which operates at best at milli-second rates.

So, I am going to be talking both about cognitive science and artificial intelligence, but I am going to be talking especially about the ideas that have come out of cognitive science, out of our research on human thinking, and the implications these may have for the future of artificial intelligence. In the past, of course, these two fields have borrowed back and forth, each one giving strength to the other over the last thirty years.

Programming Languages

Let me start with the topic of "Programming Languages" and see what we have accomplished up to the present time. The very existence of cognitive science and AI depended on having programming languages that would allow complex and irregular structures to be stored in memory. We first had the invention of list processing languages more than 30 years ago, followed a few decades later with production system languages like OPS5, and more recently, by logic programming as exemplified, for instance, in PROLOG.

The current interest in connectionism and parallel networks is sure to spawn, and is already spawning another class of programming languages. Early examples of that can already be seen in some of the programmes being talked about at this meeting.

So, today, we have these collections of list processing languages, of production systems, of logic programming languages, and of connectionist or neural network languages.

Hardware Limits

On the hardware side, of course, we can't have cognitive science or artificial intelligence at all without having powerful computers. What's less obvious is whether the availability of hardware has really been a major determinant of the speed at which the research has advanced. Has hardware been the bottleneck or, has it always or usually been available when needed, with the required memory capacity and operation speeds?

I think we have to give a mixed answer to that question. Current systems for visual or auditory pattern recognition, chess programs and some other expert systems couldn't operate at tolerable speeds with the computers available as recently as five years ago. In this sense, the remarkable and continuing advance of the speed and memory capacities of hardware has been absolutely essential to the development of AI and cognitive science.

But we can ask a different question. Has hardware development been the bottleneck that has limited the production of ideas in cognitive science? And here, I think, the answer is largely negative. The rate at which machine intelligence has been pushed into new domains has depended mainly not on hardware but on the ingenuity of researchers. Basic research, after all, seldom requires programs that have to perform in real time. We can get our ideas about how those programs should be organized, and wait to the time when computers will be able to execute them as rapidly as we need.

Now, there are exceptions. In designing programs to play chess under tournament conditions, machine speed is of the essence. Some of us have long believed that computer chess research should put more emphasis on incorporating chess knowledge in the programs, and less on speeding up brute-force search. After all, the strongest chess players in the world today are still very slow human beings who seldom look at more than a hundred branches in the game tree, while computers are looking at 3 million or 5 million branches. However, the history of progress with computer chess does not really support my argument. The history of progress has been that chess machines have gotten stronger as their equipment has gotten faster.

A few years ago, the idea was popular that AI programming would be greatly facilitated by the availability of special LISP machines or PROLOG machines. Those machines now exist, and they do achieve a speed-up, but only that. They allow us to execute important primitive operations more rapidly, but they still compete with powerful, general-purpose hardware. And the verdict is by no means clear as to whether special purpose machines will continue to be cost-effective. At any rate, they represent not so much a breakthrough, as just another source, one source of very many, of speed-up in hardware.

In the case both of PROLOG and of languages for connectionist programming, it's widely believed—and I see many expressions of that belief in the papers being given at this conference—it's widely believed that major problems of execution speed would be solved if we had massively parallel hardware, and much effort, of course, is now being devoted to producing such hardware.

I am skeptical of this belief on two scores. First, I am skeptical that parallel hardware is the answer to exponential explosion of search, which is a problem that plagues PROLOG among many languages. I am skeptical that it is feasible even in principle to design parallel hardware that has genuine general purpose capabilities. So, in my remarks, I am going to make some strong distinctions between parallel hardware that is adapted to special purposes or special classes of problems, and parallel hardware that is alleged to be useful for quite general purposes. I will have more to say later on those points.

I would simply observe at the moment that some very impressive special purpose parallel hardware has already been produced, for example, array-processors; and second, that I am not aware of any convincing demonstrations of massively parallel—that is, with many many parallel components—general purpose hardware. The newer super-computers, with only a little parallel capacity but offering the fastest computation that's now available—those super-computers are used mainly for numerical analysis. And except for connectionist research, they have found relatively little use in artificial intelligence or cognitive science today.

The significance of this observation is that we certainly can achieve major speed-ups with parallel hardware adapted to special uses, but we don't, at the present time, know how to bring about any speed-up of several orders of magnitude with general purpose parallel hardware. And it is not at all clear that parallel systems are the key to progress in cognitive science.

Now, it is clear that on this point I am challenging one of the main assumptions in many papers in this conference, and I am going to have to say more about that before I close.

Intelligent Programs

But for the moment, let me turn to computer programs. For this audience, I don't really have to list the many domains in which computer programs already exist that reach or surpass human levels of intelligent behavior in those domains. Nor do I have to list the many answers we have gained to our questions about how the human mind manipulates symbols in thinking and problem-solving, and the like.

I would only like to summarize some of the common characteristics of programs that seem important to defining the very nature of intelligence.

First of all, we do achieve speeds in computer programs that are simply unattainable for people—speed in arithmetic being the most obvious example. Nevertheless, we have found that speed and brute force, unless combined with heuristics, with rules of thumb borrowed from our understanding of human cunning, doesn't go very far toward achieving intelligence.

Very early in research on artificial intelligence, some of the rules of thumb, some of the heuristics, were discovered that permit people to search very selectively in problem spaces, that would otherwise be far too large for human computational capabilities. Even rather simple hill-climbing techniques, which select the next step with the aim of increasing an evaluation function, have proved successful and powerful in reducing exhaustive search. More sophisticated, and very widely used in expert systems, is means-ends analysis; which guides search by comparing where the system is at a given time with the goal, with where the system is trying to go.

These and other search heuristics were largely found through research on simple problem solving systems. But on problems that nevertheless can be quite difficult for human beings.

The intelligence of experts, on the other hand, is usually applied to domains that have a large information content. We know today that a world-class expert, in every one of the dozen or more domains that has been studied carefully, bases his or her expertise on possessing vast knowledge, as well as on the ability to do means-ends analysis or other forms of heuristic search.

Typically, a human expert knows 50 thousand or more things;—we call them "chunks", usually;—in the domain of his expertise. This human knowledge is stored in production-like form, in an indexed encyclopedia. I think we have excellent evidence now of this form of memory storage in the human expert—an indexed encyclopedia with about 50,000 or maybe 100,000 entries.

For example, each one of us is an expert in his own native language. In our native languages, we can all recognize 50,000 to a 100,000 words (words, not KANJI); and can immediately retrieve from memory our knowledge of the meanings of those words. Doctors do the same thing with medical symptoms. Chess masters recognize features on the chess board in the same way. And we know, because we have done it, that we can build expert systems that are capable of performing at the level of human expertise by constructing such encyclopedias in the form of production systems, and endowing them with a little capability to do means-ends reasoning or some other form of inference.

Today, we also know that the responses of human experts that we call "intuitive" or "judgemental"—where we say someone solves the problem by intuition—that these responses are simply acts of recognition based on seeing one of the 50,000 chunks or cues in the situation before the expert, and responding to that with information gathered from memory.

Knowing that—knowing that human expert knowledge is structured in this way, we have been able to push our computer explorations into the domain even of ill-structured problems and creativity. Programs have been constructed—like EURISKO, by Douglas Lenat, BACON and KEKADA, of our group at Carnegie-Mellon—that can discover laws and data and can create new concepts out of old concepts, and that can plan sequences of experiments aimed at achieving some research goal.

Most of the achievements of cognitive science, however, up to the present time relate to the programming of relatively well structured tasks, where the goals and operators are fairly well defined. More recent successes like those with the programs on scientific discovery, and even programs that compose music and programs that make creative drawings (I am thinking particularly of the English artist, Harold Cohen) raise our aspirations for research on ill-structured problems. The tasks performed by such programs involve vaguely defined goals and no clear boundaries for the legality of what we would call "moves." So, there are no longer, if there ever were, clear limits to the kinds of human thinking that can be analyzed by the methods of cognitive science, and that can be automated by artificial intelligence.

Thinking versus Perception

When cognitive science and AI began, most of us thought that it would be easiest to write programs to do the every-day things that every-day people do, simple things, and that it would be very hard to simulate the higher flights of the human mind, the kind of things that professors or engineers or doctors do. But in fact, it has turned out to be exactly the opposite.

Writing computer programs to imitate what the human eyes do, the human ears do, the human hands do, such programs have been much more difficult to write than programs to do the things that deep thinkers do, or that engineers do, when they are solving mental problems.

Now, we really should have predicted that; it shouldn't have come as a surprise as it did. We should have predicted it, because the sensory and motor systems, which we have and share with the mammals, have been evolving for hundreds of millions of years. In the course of evolution they have become highly sophisticated, highly tuned devices. And it should not be surprising that it is hard to simulate them.

The new human brain, the part that does language, the part that does abstract thinking, is in fact a very newly evolved device—hardly a million years old. And we should expect it to be, as it has turned out to be, far simpler than the sensory and motor systems.

Learning

Let me say a word about learning. From the very beginning, cognitive science has been fascinated with learning, but in the early years—the first twenty or thirty years—had very little success in it. We always mention Samuels' checker program, because it's about the only example of successful learning in the early history of AI.

In the past decade, learning has taken off again, and much has been accomplished. One of the most significant accomplishments is the understanding we have gained about how human beings can learn to solve problems by examining worked-out examples, and then re-programming themselves to retain the skills in those examples.

The computer counterparts of schemes for learning from examples are what we call today adaptive production systems, which are simply production systems that can themselves form new productions and store them in memory where they become a part of an augmented program. Beginning with the work of Neves a decade ago, it has been demonstrated convincingly that such systems can be built for learning subjects like algebra or geometry at the high school level. In fact, an experimental program is going on today in the Beijing public schools, where children are being taught algebra and geometry in the standard high school course with the use of the technique of learning from examples, without lectures from a teacher. And that experiment is going very successfully.

Applications

A little later I am going to talk about quite a different approach to learning, that's very popular today, namely, connectionist learning schemes. But let me say one more thing about where we are today before I get on to the research frontiers: something about applications.

I have already mentioned some of the main areas of applications of cognitive science and AI today, including the whole set of things we call "expert systems." Among other real world applications, which really are just at the edge of application even today, robotics has had a great deal of visibility. But I think most of us realize that the bulk of the robots actually working in factories today are based on quite traditional control theory techniques, and that artificial intelligence is just beginning to have an impact on robotics. That impact depends on the progress, very slow to date, that we are making in developing sophisticated sensory and motor devices.

We now understand a great deal about how humans process natural language and are beginning to bring more and more systems into application that make use of various, still limited but genuine, natural language capabilities.

The real problem that stands in the face of our progress on natural language is that a language translator—a system that could go from one language to another, or from non-linguistic objects to language, or from language to non-linguistic representations—must have a great deal of semantic knowledge about the subject matter on which it is translating. We have had ample evidence from the experiments of the last thirty years that syntax is not enough, that systems that have sophisticated syntax get us only a very short way, and to go the next step into systems that have much broader application, we are going to have to take seriously bringing in huge amounts of semantics.

The Future

This very brief sketch of where we are today in cognitive science forms the basis for the forward view I would like to present now from these frontiers, and I think, will enable me to be rather brief in describing the prospects in the years ahead.

First, I want to say something about the areas that I have already identified as critical. Then, I want to say something about our needs for software and hardware supporting systems, and our prospects for meeting these needs. You have already seen that my views on parallel computing may be rather different from some of the views presented at this conference. You will also see, as I proceed, that my views on logic programming are rather different from the views that are being presented by most people at this conference.

So, please, before you condemn me as a hopeless reactionary, I hope you will listen carefully to the reasons for my position, and then we can have an intelligent discussion about it.

Important Domains for Progress

Let me first say something about a few task domains and the directions I see us moving, and I will refer again to robotics, to language, expert systems, learning, and representation. I will only have to say a few words about the first of these, because I have already made my main points.

Robotics

With respect to robotics, the main point for the near future, the next decades, is that we are going to have to focus our efforts on the development of adequate sensors and adequate effectors for our robotic systems, and adequate feedback from those sensors—adequate feedback from the robot to the planning system, so that the planning system can readjust its thoughts to where it is really standing in the world. It can readjust to reality periodically.

This calls for a great deal of very hard and detailed work. I see no magical breakthrough. All of us will be happy if it comes; I don't see what direction it is likely to come from. My crystal ball does not have any robotics breakthrough in it.

Now, connectionism may play an important role in this, it's clearly in the sensory domain that we can most likely make immediate or near-future use of parallel capabilities of a connectionist sort or of any sort. The evidence from psychology is extremely strong that most human higher mental functions—the thinking that takes place in the central nervous system in the new brain—are carried out in a serial, one-at-a-time fashion; with all of them passing through the narrow bottleneck of attention.

If you still have any doubts about that, you can very cautiously perform an experiment by going out with a friend in a car, in Tokyo traffic, starting a conversation with the friend and watching what happens to the conversation as the traffic gets denser. And I hope your friend will give first priority to the traffic, second priority to the conversation. Human beings simply do not perform multiple tasks that require attention in real time. We are serial systems.

Now, it is just as clear, from the psychological evidence, the evidence of the laboratory, that the eye and the ear, and to a lesser extent, the motor system—I am distinguishing them now from the so-called "higher mental processes"—that these peripheral devices are in fact parallel devices. And it is here that the main connectionist research effort, it seems to me, needs to be focused. Some connectionists think that all cognitive problems can be handled with their models, without the need for a separate symbolic level. That may turn out to be true, but for the short run, I think, they can perform much more useful work by simulating effective sensory organs, and to some extent, motor organs.

There is more to robotics than sensory and motor systems. There has to be a thinking and planning system to connect them. Much of the basic equipment and organization for that is in place in systems as old as STRIPS, which was derived from the work on problem-solving. But the problem that needs more attention and is just beginning to receive it is how a planning system using a very gross and inexact model of the world outside guides a robot that has to survive and operate in that real world. So, we need to solve the problems of correction and feedback of the planning models.

Language

With respect to language, I have already spoken about the need for pushing forward on the subject of semantics. Douglas Lenat and his colleagues in Texas are engaged in building an information base of encyclopedic dimensions that can be employed to test the use of semantic knowledge in informing and guiding language understanding and translation systems. I would expect and hope to see more enterprises of this kind, guided by what we already know about large expert systems, about production systems, and about data base architectures.

If there are any fundamentally new ideas that have to be invented to push that particular part of the work forward, they are not visible to me. Undoubtedly, new ideas will emerge as the work advances. Intelligent empirical work always produces new ideas. But what's needed right now, and in the near future, is large-scale—I mean, really large scale, I'm talking of millions of items—large-scale experimentation with data bases.

And for this particular approach to language processing, we probably need effective big memories more than we need fast computing. The real parallel capabilities in the central part of the human nervous system are memory capabilities, not parallel processing capabilities.

Expert Systems

I have nothing special to say about the needed emphases in expert systems. I think the development of expert systems will follow as a byproduct of the work in these more fundamental areas that I am talking about.

Learning

With respect to learning, there are two or three main foci of learning research today. I have already mentioned the two that seem to me most promising. One is connectionist research for the learning of visual and auditory patterns. The other is research on adaptive production systems that learn from examples. I am sure that those don't cover the whole waterfront of learning programs, the whole range of mechanisms that the human brain uses to improve its performance. But they do seem to be among the most important. And we understand enough about them today, so that I think the research on them will move ahead quite rapidly.

A very intriguing question, on which one can find many opinions and not much evidence, is when one should choose learning and when one should choose programming as the preferred method for giving new knowledge to an expert system. Of course, we humans gain all of our knowledge by learning, but that's maybe just by default. Nobody knows how to open up the box and put a program in it. That might be far more efficient than going even to a good university. We don't know how to do it.

In the case of computers, we have the option. We can try to program computers, or we can try to get computers to learn. I think an interesting question we could ask as an exercise, if we are interested in learning programs for computers, is whether in fact we would program people instead of teaching them if we could. Maybe we should be very selective in our goals for learning by computers. Maybe we already have a better way available to us in programming.

I am sure that's only part of the story, but it's a question worth asking before we plunge into large research projects on applications of learning to computers.

Representations

Then there is the important and difficult topic of representations. Information, if we are to process it, has to be taken out of the abstract world of ideas. It has to be given some concrete form of representation before it can be processed either by computers or brains—some pattern of electromagnetism in a computer, some pattern of neural activity in a brain.

Today, the typical representations we use in computing are list structures, schemas or whatever you like to call them;—scripts, frames, whatever;—I prefer to call them "schemas." All of those forms are good for stating propositions, or can be made to look as though they were stating propositions. But there is a great deal of evidence from the psychological laboratory that we human beings use pictures or diagram like structures as preferred representations in much of our thinking. We don't think in words. Einstein always claimed that his serious thinking was not done in words, but in some kind of picture-like, fairly abstract but picture-like structures. We are just beginning to ask what these non-propositional representations might be, ways of representing information other than by stating propositions, and how these representations can be implemented in computer systems and simulated for cognitive research.

I will refer to one early effort along those lines which still hasn't really been surpassed, Novak's ISAAC program, which was announced, I believe, at the IJCAI Meeting in 1977, a program which understands physics problems stated in words, then writes the equations for the problems and solves them. But ISAAC did that not by translating syntactically directly from words to equations; ISAAC does that by translating from words to diagram-like structures stored in the computer. We know they are diagram-like, because the diagrams can be drawn by ISAAC on a CRT—the structures can be interpreted and exhibited as diagrams of the scene. Then, those diagrams are used to construct the equations. The process that ISAAC uses, however primitive it is, looks very much more like the processes that we human beings use in solving problems that are presented to us in words, or problems we encounter in the real world, than systems that make use only of abstract propositions or equations.

It is sometimes said that a problem well represented is a problem half solved. That may be an exaggeration. There are lots of problems that you can represent well and still not solve, Fermat's last theorem being a good example. The mutilated checkerboard problem, which was introduced into AI by John McCarthy (the problem of covering a checkerboard with dominoes, each domino covering two squares, after you have removed two diagonally opposite squares of the checkerboard) is an exceedingly hard problem for human beings. A computer could solve it exhaustively in a very short time by trying all possible coverings, only a few tens of thousands. Human beings don't solve it that way at all. A human being solves it by changing the representation, by ignoring after a while the checkerboard itself and replacing it with an abstract representation of the numbers of black squares, the numbers of white squares, and the numbers of dominoes. And then, the problem becomes very easy.

So, there are problems whose solution depends on finding the right representation. And that representation is not by any means always a propositional representation, although in that particular case, it is.

I will come back to the representation issue in my final remarks, but first let me say something about the supporting systems, both software and hardware, that we need in order to build the sort of structures that I have been talking about. And I would like to do this in terms of four basic issues.

Serial and Parallel Systems

First, I would like to say more about the serial and parallel architecture issue. I would also like to say something that's connected to that, about connectionism. I would like to say something about logic programming, and about the non-verbal representations that I just introduced.

Let me then take a broader look than I have in my previous remarks, at the serial versus parallel issue. Clearly, the human brain is a vast network of neurons. Some people think there are about 10^{12} neurons. The number doesn't matter; it's very large. Since those neurons are all there, it's very natural to think that intelligence must require parallel computation. Why else would nature have built this tremendous processor that looks so parallel?

But the matter isn't quite so simple. First, our computers also have always been parallel devices. Von Neumann computers are very parallel devices, if you include as parallel the enormous memories that they have, all holding information in parallel. True, memory is a static component, or relatively static component, of the computer. But perhaps, those parallel components in our brain are also largely static. There simply is no neurological evidence, for most of them, as to what role they are playing.

Second, I pointed out earlier that the human thinking process contains a narrow bottleneck of attention, which severely limits the number of ideas that can be entertained at once. The human nervous system is also an extremely slow system by computer standards. A simple act of recognition, recognizing your very best friend as he comes walking down the street, takes the better part of a second.

The slowness and seriality of the brain have made it possible to simulate such activities as problem-solving and language understanding in considerable detail using general purpose serial computers. And there is no doubt that where we have done it, modern computers can run much more rapidly than human beings in tasks that do not require the operation of basic sensory recognition capabilities.

On the other hand, we have seen that, if conscious thought is demonstrably serial, seeing and hearing are demonstrably parallel. So, we don't have to take an either-or attitude on this serial-parallel debate.

It seems to me that there is lots of room in the future of artificial intelligence for the kinds of serial devices, including the von Neumann architecture that we have had to date. They will continue to do a very important part of our computing, and we should ask what the special mission is of the parallelism. I have already indicated what I think that is. What I haven't indicated is what the right kind of parallel hardware is. And of course, if I knew the answer to that, I would be designing a parallel machine somewhere, and I think many of you are working very hard to find the right answer.

Anyone who has attempted it will testify that achieving massive parallelism in computation is extremely difficult, except where the hardware is custom designed to handle certain special kinds of precisely defined tasks. If you have arrays to deal with—very homogeneous arrays, then, array processors will operate in parallel very rapidly and very efficiently.

But general purpose parallel processors, going back to ILLIAC IV and its ancestors and descendants, have proved to be enormously hard to program, except for tasks whose precedence requirements match closely the hardware design. And we know, for hardware being produced nowadays, employed on tasks that are not closely matched to the hardware, a typical expectation is that you may achieve a speedup of a factor of three to five, with the use of 30 processors.

There is no reason to believe, that someone will invent a clever idea that will suddenly make general purpose parallelism feasible. The difficulties aren't superficial; they are fundamental. Parallelism is constrained by the precedence requirements of the tasks that we expect to compute. When there is little connection among the tasks, then, we can have a lot of parallelism. We can get a parallelism, if there is no connection among the tasks, by putting 500 computers in the room and giving each one a different task. That's parallelism, too. We know how to solve that problem.

But the minute you get dense and rigid connections among the tasks and difficult precedence relations, then a large part of the potential capacity of the parallel machine goes unused unless the parallel machine is adapted to just that kind of task.

There is a very interesting description, some of you have probably seen, or will hear about later this week, of Felton and Otto's highly parallel chess program, using 512 processors in NQ. That program, very ingenious, is no exception to what I have been saying. If you read their article carefully, you will see what pains had to be taken to take account of the special requirements of the chess task to get even 40 percent efficiency out of the 512 processor machines, a very high level of efficiency. That only is attainable if each of the subtasks requires a considerable number of seconds for their execution. If each of the subtasks takes several minutes for execution, you can get a very high utilization of that machine. The utilization drops off rapidly if you are dealing with very short tasks and very rapid interchange of tasks. This is typical of the problems of parallelism even when attacked with very sophisticated software.

Elsewhere I have speculated as to why natural structures, the kind of structures we see in nature, complex structures, have evolved mainly into hierarchies:—Atoms forming molecules, molecules forming macromolecules, macromolecules forming cells, and so on. Why do we have these hierarchies? This evolutionary lesson, that hierarchies seem to be the fittest form of complex systems, is one that designers or computer architects might examine closely and consider imitating.

Of course, we already have had considerable experience for the hierarchical organization of memories, but very much less experienced with hierarchies of active processors.

The conclusion I would draw is that we will continue to make progress toward parallelism, but probably without a sudden burst of success for a general purpose parallel machine. In fact, parallel architectures designed with particular applications in mind, like vision, are likely to advance more rapidly and to reach more satisfactory levels than attempts at general purpose, massive parallelism.

And meanwhile, until we learn more about how the brain really operates, and what part of it is really parallel, the design of hierarchical systems deserves more attention than it has received.

I have already stated reasons for thinking that connectionist systems as a form of parallelism might play a large role in modeling sensory and motor systems. Again, I would ask at what point those connectionist systems need to be built into more hierarchical form instead of trying to operate at a single level?

The observation of hierarchy throughout nature is a reason for thinking that the mind is arranged in levels, that there is a level of neuronal organization, and that these neuronal systems, in turn, implement the primitive structures and operators of the symbolic systems at the next level above.

Research, after all, is an exploration into the unknown, not into the known. As far as research programs are concerned, a good philosophy is to let a hundred flowers bloom. Both connectionist and symbolic directions of research hold out great promise, and there is no urgency today to draw exact boundaries between their respective spheres of applicability. But in particular, connectionists should be encouraged to give high priority to the problems of processing sensory stimuli.

Logic Programming

Now, let me say something about logic programming. The analogy between computing and logical inference has a long and an interesting history. Of course, it really started the other way around. Aristotle modeled logic on human reasoning, and Turing modeled it on a computing machine.

Now, let me take up the specific topic of PROLOG as an example, but only as an example, of a logic programming language. Simply put, the idea behind logic programming is that reasoning should be logical;—who could deny that? Well, I am going to deny it in the moment. The idea is that programming languages should incorporate from logic the principles and the insights that make logic a powerful and rigorous form of reasoning. Underlying any inferential system are principles, some of which are expressed in declarative form, others in procedural form. The declarative ones are called axioms; the procedural ones are called inference rules.

Formal logic, as it comes to us, has always had the ideal that both axioms and inference rules should be independent of subject matter—they should be analytic or tautological. They should give valid results for all possible worlds. Then, when the logic is applied to a particular domain of thought, additional axioms—domain-specific axioms—are supplied to specify what's known about that domain.

Moreover, formal logic has always been closely connected with questions of rigour in reasoning. Systems of logic are usually designed to allow you to verify that things are correct. This is accomplished, first, by separating the logic axioms, as I said, from the domain-specific axioms and second, by limiting the inference rules that are to be used. By using axioms rather than inference rules, Whitehead and Russell have got along simply with modus ponens and substitution, and no other inference rules.

A very heavy price is paid for adhering to these principles. The reasoning proceeds by tiny steps, huge numbers of which are needed for even the simplest proofs. We certainly found that out in computer theorem proving. Why does that field moves so slowly? Because we have insisted of putting all the power into axioms, and have used a very limited range of inference rules, usually, resolution principles of one sort or another.

The slow, and to some of us disappointing, progress in automatic theorem proving by computer provides evidence of the cost of adhering to the principles of logic at the expense of alternative possibilities. Only grudgingly did the authors of early theorem proving programs admit such obvious inference procedures as equality, commutativity, transitivity. The earliest theorem proving schemes actually axiomatized those things, and spent all of their time proving equality or proving commutativity, instead of doing the real work that they set out to do. Today, single rules of inference like resolution and its derivatives, as in Horn Clause resolution in PROLOG, are still generally preferred over systems with multiple rules.

Now, when we examine human reasoning, in the psychological laboratory or in real life, we see that it proceeds in quite a different way than the way of the logician. It uses not just a few inference procedures, but many, and these are not all logic rules, but generally incorporate important domain-specific knowledge. If we watch a good student solving a problem in kinematics, we find the law of uniform acceleration is being used not as an axiom but as a computational procedure for inferring, say, distance from time and acceleration. The human processes in situations like these are readily modeled by production systems with relatively little use of declarative knowledge—almost the opposite of the principles that guide logic programming.

Human reasoning is a mixed bag which serves many purposes. It's used to a much greater extent to discover than to verify. And we know that discovery often requires heuristic search: taking long jumps often at the expense of guarantees of completeness, at the expense of guarantees of validity. Now, the lack of those guarantees is not a virtue. It would be nice if we could do our problem solving at the same time we were sure that our system was complete, or that every step was guaranteed to be valid.

But this is the price we pay for living in a world where completeness and guaranteed correctness are almost always computationally infeasible—unreachable. It is better to find an answer sometimes, than to be sure that you will eventually find it, if "eventually" is long after your death. Better to check after you've found a candidate than to refuse to hazard possibly false steps.

Now, the principles I have just announced are not laws of logic. They are empirical generalizations from human experience. In most real life situations, human reasoning is, and must be, heuristic search, using rules of thumb. If powerful inference rules, even vulnerable ones that are not always correct, can be incorporated in the search, it will be more likely to reach its goal in a tolerable time.

Now, of course, there is no reason in principle why logic programming cannot be carried on in exactly this spirit, just as there is no reason why a language like PROLOG can't be extended to equivalence with a Turing machine. By giving up all of its formal properties, making liberal use of the cut, adding all sorts of auxiliary functions to PROLOG—you can get PROLOG or any other programming language to do anything. But if you program a logic programming language in the spirit in which it was designed, if the principles of logic programming are followed, then, logic programming loses its special rationale and claim to preference.

Contrary to the underlying justification for logic programming, effective computing procedures have to be substituted for declarative statements and flexible first-best search has to replace depth-first, back-track search.

My problem isn't with a programming language, but with a misconception of the central principles that underlie intelligence, and that should guide the design of intelligent programs for AI. Among those central principles is the idea that problem-solving is after all heuristic search.

One of the oldest issues in cognitive science is whether knowledge should be represented declaratively or procedurally. As in the serial-parallel case, the answer undoubtedly is "both." There is probably good reason to believe that much of our knowledge of the world is stored in declarative form, but that much of it is stored procedurally. We need to be suspicious of proposals to place the whole of intelligence, or nearly the whole, in one or the other of these forms of representation, or, I should add, even in both of them.

Because in my plea for a balance between these kinds of knowledge, I haven't said what I mean exactly by "declarative representation." And I would not want to identify that with "propositional." I would want to include the kinds of diagrammatic and pictorial representations that I mentioned earlier. Because list structure memories can be used not only to represent propositions, but to build representations that are computationally equivalent to diagrams. I refer again to Novak, or to the article that Larkin and I published in Cognitive Science in 1986 on why a diagram is sometimes worth ten thousand words.

If it is true, as seems probable, that much human reasoning uses picture-like and diagram-like mental and external representations, then research on computer hardware and software for implementing such representations will be of great value. I think it is being relatively neglected in the research going forward nowadays. There has, of course, been substantial research activity of this kind in connection with computer-aided design, CAD. But to the best of my knowledge, it has not been closely linked with research in artificial intelligence or cognitive science. A closer linkage could lead to very interesting and useful ideas about how to represent knowledge that is declarative, but not explicitly propositional.

Conclusion

From the very beginning of research in AI and cognitive science, researchers have been accused of excessive optimism—the nay-sayers, those who are sure, a priori of its impossibility, have accused researchers in artificial intelligence of being too enthusiastic. I hope we have been guilty of some optimism. No field goes forward without optimism.

In a field that has moved as far and as fast as cognitive science and artificial intelligence have in the past thirty-five years, I think we have lots of grounds for the optimism that we express. Our understanding of both human intelligence and machine intelligence is large today; it continues to widen and deepen at a rapid pace. If there are any limits to the kinds of intelligence that can be represented by computer programs, those limits have not yet made themselves evident.

If I have been skeptical this morning that we need anything that's properly described as a breakthrough before we can proceed further, I am not skeptical about the research possibilities for important new ideas and important advances.

We human beings, over the centuries, have been fascinated by four great questions: The question of the nature of matter—what is matter all about, how do we create a world out of matter, the things that high energy physicists study; the origins of the universe which we are probing today through astro-physics—the big bang, and what was there before the big bang; the nature of life, which has been so much advanced by molecular biology in our lifetime. And the fourth question, the emergence of mind from matter; the mind-body problem. How does anything like a brain, a meat machine as Marvin Minsky has called it; how does a brain accomplish thought; how does anything as material as a computer, made of metal and glass, accomplish thought?

Until the computer was recognized as the general physical symbol system that it is, we had almost no tools for investigating the nature of intelligence and mind. We in this room, our generation, are the fortunate ones, who have been alive just at the time this powerful tool has become available to aid us in our research.

Combining its intelligence, the intelligence of the computer with our own, we will continue to move rapidly toward a fuller and clearer conception of the minds of both computers and people.