CS protein can protect against malaria in animals such as mice. One monoclonal antibody, Ruth Nussenzweig says, "completely inhibited the invasion of the liver by interfering with recognition and penetration."

In the more recent work, both cloning groups have found that short synthetic peptides containing the repeating tetrapeptide block the binding of monoclonal antibodies to the sporozoite protein, results which show that the antibodies are specifically directed against the repeat. In addition, the NYU workers have shown that monoclonal antibodies against the repeating peptide of *P. knowlesi*, which is 12 amino acids long, bind to sporozoites of this species and abolish their infectivity.

The results suggest that it might be possible to immunize humans against malaria by using either synthetic peptides or the whole CS protein. But, Miller notes, a very high level of immunity will have to be maintained to inactivate 100 percent of the sporozoites. If even one escapes unscathed into a liver cell, the disease can still occur.

Work on an antisporozoite vaccine is only one aspect of current research on malaria prevention. A great deal of effort has been directed at identifying antigens from the merozoite and red blood cell stages of the parasite that might prove effective for vaccination. In general, these stages are more complex and variable in their antigenic composition than the sporozoite.

Nevertheless, several laboratories have been making progress in cloning genes that might be used to vaccinate against these forms of the parasite. One such gene appears to code for the protein by which the merozoite recognizes the red blood cell. This protein may be a good candidate for a vaccine. It is effective in evoking an immune response and, because it must interact specifically with the human red blood cell membrane, may be less subject to variation than other merozoite antigens.

An effective antimerozoite vaccine would prevent or ameliorate symptoms in an infected individual, but would not block further spread of the parasite. An antigametocyte vaccine, which is also in the works, would prevent the spread of malaria without affecting the course of the disease in the currently infected person. "It's the same as killing off the mosquito," notes Cross.

The different approaches to a malarial vaccine all have different strengths and weaknesses and it is currently difficult to say which will ultimately pay off. It is also possible that more than one type of malarial vaccine will be needed to give effective protection. As Cross sums up the current situation, "The science is in a very exciting stage, but whether it will lead to clinical results is unknown."

—Jean L. Marx

Artificial Intelligence in Parallel

Working with many processors at once could accelerate computation enormously—and suggest new ways to think about thinking

It is a painful fact that modern computer intelligence is sharply constrained by the computers. "The bigger we make our programs, the 'smarter' they get and the slower they get," laments Thomas Knight, a researcher in artificial intelligence (AI) at the Massachusetts Institute of Technology (MIT). "We're in the embarrassing position that we give a program more information, and it gets worse."

Consider expert systems, for example, which are programs that try to encapsulate human expertise in a set of rules. The available processing power sets a practical limit of just a few thousand rules, which means that even the best artificial expert is nothing more than an idiot savant in one narrow area. Getting computers to exhibit "common sense" in real time means speeding them up and increasing their memory capacity by a factor of thousands or millions.

Or consider one recent experiment in which a computerized cart successfully used a vision system to pick its way through obstacles in a hallway: every time it advanced a meter it had to stop and reassess—for 15 minutes. Getting a system to walk and chew gum at the same time, so to speak, means speeding up this kind of performance by a factor of roughly a million.

The irony is that the brain is beating out the computers with neurons that operate about a million times slower than silicon. And the secret, of course, is in the wiring: the neurons are in there doing millions or billions of operations simultaneously. Whereas computers, with few exceptions, are still based on the serial, one-step-at-a-time architecture laid out by computer pioneer John von Neumann in the 1940's: one central processing unit, one memory bank, and one data channel connecting them. It is rather like having one bank teller on duty during the lunchtime rush.

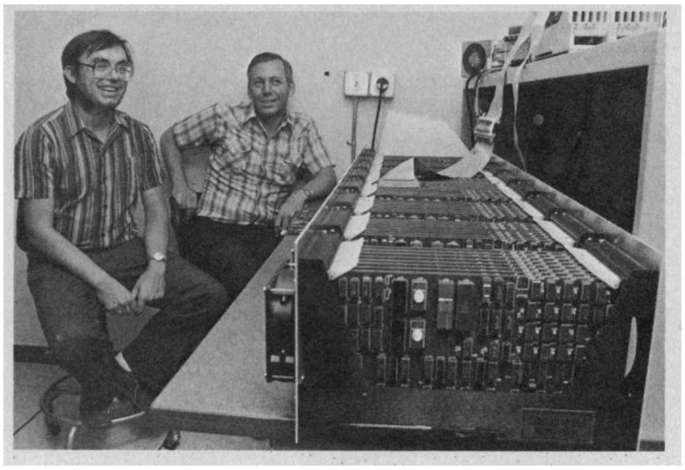
Thus the current rage in the AI community, and for that matter in the computer science community generally, is "parallelism." The idea is to build machines with many independent processors doing many things at once—perhaps even several million things at once.

Actually, this is a new revival of an old idea. People were proposing and designing parallel computers as long ago as the 1940's. And starting in the 1970's, as advancing technology began to make it possible, high-performance supercomputers began to incorporate certain kinds of concurrency through the use of "array" processors and the "pipelining" of data. The current generation of supercomputers can handle as many as 16 data streams simultaneously.

But the striking thing about the current ground swell is its emphasis on massive, million-processor parallelism. A big part of the push comes from the ferment in AI applications. For example, the Defense Advanced Research Projects Agency (DARPA)'s Strategic Computing Initiative has given top priority to the development of very fast symbolic processors. So has Japan's Fifth Generation project (*Science*, 24 February, p. 802). Moreover, both military and industrial users of AI are putting a premium on real-time performance in such things as robot vision and natural language interfaces.

But a large part of the interest also comes from the possibilities being opened up by semiconductor technology. It happens that many of the parallel designs are well suited to very large scale integration (VLSI). And by no coincidence, it also happens that DARPA has had a VLSI program since 1979: simply send in a design to one of the DARPA-sponsored "silicon foundries" and back it comes as a chip. The result is that lots of people are out there trying lots of things.

The upshot is that about a dozen parallel machines are now under development for AI alone, plus at least that many more being developed for scientific and technical number-crunching. As might be expected, the approaches are diverse—there is essentially only one way to do a serial machine, but a near infinity of ways to build a parallel machine—yet there are a number of fundamental issues in common:


Caltech's cosmic cube

Sixty-four "IBM PC's" linked in parallel give one tenth the power of a Cray-1 supercomputer for 100th the cost. Developers Geoffrey Fox (left) and Charles Seitz look on.

• "Granularity": That is, how big and how capable are the individual processors? The number-crunching designs in the scientific domain tend to use just a few fairly large processors, mainly because the problems they address tend to have a predictable and highly repetitive structure. An example is the Cosmic Cube being developed by Geoffrey C. Fox and his colleagues at the California Institute of Technology. The $80,000 Cube uses 64 boards, each roughly equivalent to an IBM Personal Computer, and has about one tenth the power of a Cray-1 supercomputer at 100th the cost. It has already shown its worth in quantum chromodynamics calculations (*1*).

A somewhat finer-grained approach from the AI domain is DADO, a product of Salvatore Stolfo and his colleagues at Columbia University. DADO is a "rule processor," specialized for running expert systems and similar knowledge-based programs. The latest version, due for completion in late 1984, will incorporate 1023 individual processors and will grind through rules about ten times faster than conventional machines.

More generally, however, the problems that come up in AI tend to involve small sets of data examined in many different and nonpredictable ways. The classic example is the sequence of moves in a chess game. So the AI builders, perhaps inspired by the massive parallelism of the brain, tend to emphasize processors so fine-grained that whole banks of them can be etched into a single VLSI chip.

An example is NON-VON, being developed by Columbia's David Shaw—as it happens, right across the room from DADO. The ultimate goal of NON-VON is one million processors, each with a tiny sliver of memory. Or alternatively, says Shaw, one could view the array as an "intelligent memory," in which each small group of cells is capable of doing a little processing.

As the name suggests, Shaw and his colleagues hope that machines like NON-VON will someday provide a reasonably general-purpose alternative to the von Neumann architecture. They have done simulations to show how the NON-VON approach would work in data-base management, in rule-based AI systems, and in computer vision. They have a three-node prototype of NON-VON already working, using custom VLSI chips, and they hope to have a 128-node version running by the end of 1984.

On the other hand, Shaw also points out that neither NON-VON nor any other design will do everything. One of the main goals of current research is to find out just which architectures work best for which problems. "We're not sure yet," says Shaw. "We'll just have to choose a number of different areas, try them out, and see."

• Topology: How are the processors connected? Does every one connect to every other? Or just to its nearest neighbors? Are the processors arranged in a tree hierarchy, as they are in NON-VON and DADO? In a higher dimensional cubic lattice, as in the Cosmic Cube? In a ring? Again, the "best way" depends on the application and the algorithm—tempered by the realities of the thousand- or million-processor environment.

"It's turning out that communication is *the* technical problem," says MIT's Knight. Quite aside from having to cram in all those wires, the designer has to make sure that the data always goes where it is supposed to go. "Compared to that, the computations themselves are a piece of cake," says Knight. The immensity of the problem is reflected in the name of a million-processor concept under development at MIT and independently at Thinking Machines, Inc., of Waltham, Massachusetts: the "Connection" Machine.

• Control: Dividing the work over lots of processors does speed things up. But subdivision has its costs, not the least of which is the cost of coordination.

In a centralized control scheme, which is the most straightforward extension of the von Neumann architecture, a "foreman" passes out instructions through some hierarchy arrangement to the processors that actually do the work. This does force all the processors to operate in lockstep. But the coordination is fairly straightforward and, perhaps more importantly, the individual processors can be very small and very simple: they do not have to store programs locally. This in turn means that a single VLSI chip can integrate a lot of raw processing power. The simplest version of NON-VON uses centralized control, as does the Connection Machine.

A more radical departure is decentralized control, in which bits of data are set loose in the array like taxicabs roaming Manhattan. Generally speaking, there are two approaches: in a data-driven, or "data-flow" system, the processors execute their instructions as the data become available; in a demand-driven, or "reduction" system, the processors execute instructions when the results are needed. Either way, there are obvious problems with coordination. Moreover, each individual processor has to store its own program in its own memory, which means it has to be relatively large and complex. On the other hand, decentralized architecture may turn out to be the best way of making logical inferences on large knowledge bases—a possibility that led the Japanese to choose it for their Fifth Generation project.

• Algorithms: The ferment in parallel processing has set a lot of people to thinking about how to program these machines. Does the problem break naturally into pieces, for example? If so, how? If not, are there other ways to exploit the parallelism? "It's a new kind of beast, and there's a lot to learn," says Knight. "Think about starting out with von Neumann computers in the 1940's, when we didn't even know about subroutines. We call one of our big projects here at MIT 'rebuilding computer science.' "

Given the quantities of effort going into parallel architectures, it is legitimate to ask what parallelism really buys. Some people are already grumbling about overheated expectations: "Parallel processing is one of AI's wonderful red herrings," says Roger Schank of Yale University. By itself it provides nothing but speed, he points out. It says nothing new about knowledge representation, memory, learning, common sense—any of the truly fundamental problems of AI. The fact is that any parallel algorithm can be mimicked on any serial machine, albeit slowly. "First we need to understand what we're building models of," he says. "If we could build a machine that could function correctly, but slowly, *then* parallel processing might have something to say."

Other researchers, however, while echoing Schank's point, do maintain that fast parallel processors will allow people to try things that would otherwise be prohibitively time-consuming. "It won't solve AI," agrees Knight, "but it will speed up certain key areas that are stopping us in certain problem domains." If nothing else, the rapid feedback should help hone researcher's intuition about problems. They could get results in seconds or minutes instead of weeks.

A more subtle, and perhaps more important, point is raised by AI pioneer Allen Newell of Carnegie-Mellon University: namely that new architectures, and new ways of programming, may trigger new ways of thinking about intelligence. "We aren't smart enough to change our thinking without that kind of challenge," he says. In particular, the effort may help bridge the gap that opened up long ago between AI and cognitive psychology on the one hand and neuroscience on the other.

"Back in the 1940's and continuing through the 1960's," says Newell, "there was this substantial effort to build 'neural nets'—models of the nervous system. But it petered out because there were never enough useful results.

"What did take off was AI, which has been going on in parallel," he says. "But AI was based on the building of symbolic mechanisms for intelligence, with no direct reference to the neural mechanisms. So that gap between psychology and neuroscience, which was long-standing, didn't get closed by AI. The whole thing got separated even further."

"Then," he says, "in the mid-1970's the late David Marr at MIT revived the whole idea, especially with respect to vision. He was influential, he was charismatic, and he developed a whole theory of how vision ought to happen, based on an understanding of the neural system and the psychological data as well as AI. [*Science*, 15 June, p. 1225]. That has encouraged a school of 'new connectionists' to think they can build a system to model intelligence the way the brain does."

Among the leaders of that school is Newell's Carnegie-Mellon colleague, Geoffrey Hinton. "There has been a long-standing split in AI," Hinton says: "those who say you don't need to worry about the hardware in AI, just the data structures and so forth—this is the approach that has led to expert systems and all the success in the marketplace—and a much less influential school (so far) who say that the kind of hardware available determines the kind of problems you can do well."

"Until we know how to do the proper kind of programming, we don't know what kind of system to build."

Hinton himself, for example, is interested in the representation of knowledge in the brain. "If you're in a jungle and you see a tiger," he says, "you don't just make a visual identification; you recognize a great deal more about tigers, and danger, and running." The brain, with all its billions of neurons working in parallel, does this very rapidly. It has to. Whereas a serial computer performs such operations very slowly, if at all.

Or consider human memory: "Every day, thousands of your brain cells are dying, but there's no loss of memory," says Hinton. Somehow, memories are distributed over many cells. Whereas in a conventional computer, the failure of an electronic memory cell means that a piece of data is gone forever.

Recently, Hinton and his Carnegie-Mellon colleagues have been involved in a particularly intriguing approach to simulating such behavior. Known as the BOLTZMANN architecture, it is a confluence of work by Scott Kirkpatrick and his colleagues at IBM (*Science*, 13 May 1983, p. 671), John J. Hopfield of the California Institute of Technology, Hinton and Scott E. Fahlman at Carnegie-Mellon, and Terrence J. Sejnowski of Johns Hopkins University. (A very similar approach has been independently developed by Paul Smolensky of the University of California, San Diego.)

Basically, a BOLTZMANN machine would take input data (say, a scene of a jungle), combine it with previously stored "memories," and search for the set of hypotheses that give the best match ("A tiger"). Of course, this sort of thing has been tried before, and in any realistic setting has always foundered on the vastness of the search. "It's hard even to find a *good* solution, much less the best solution," says Hinton.

But BOLTZMANN borrows an elegant trick from physics to search all its hypotheses simultaneously: first it randomizes its own internal state to achieve a kind of "thermal equilibrium," subject to constraints imposed by the input and the preexisting memories; then it lowers the "temperature" until the hypotheses crystallize around the desired best match. (The name of the algorithm honors the 19th-century physicist Ludwig Boltzmann, one of the founding fathers of statistical mechanics.)

The beautiful thing about this, says Hinton, is that BOLTZMANN will settle into a match even if the initial data is noisy or incomplete. (Say you only got a glimpse of the tiger's tail.) By the same token, BOLTZMANN could model "content-addressable" memory: an isolated input (the tiger's tail) might find a match within a much richer structure of knowledge (Tiger, danger, run . . .).

Moreover, as implemented in parallel hardware, BOLTZMANN would be very robust. Memories would be distributed throughout a network of tiny processors; the failure of one processor would thus make very little difference.

Simulations of the BOLTZMANN algorithm on serial computers have shown that it really does work. On the other hand, BOLTZMANN is still a long way from being a practical machine, not least because there is still a lot to be learned about its limits and capabilities.

"It's a paradox," says Hinton. "People are using conventional computers to simulate these parallel ideas. It's very slow, and eventually we'll have to build parallel systems. But until we know how to do the proper kind of programming, it's too early—we wouldn't know what kind of parallel system to build."

—M. Mitchell Waldrop

Reference

1. E. Brooks III *et al.*, *Phys. Rev. Lett.* **52**, 2324 (25 June, 1984), p. 2324; S. W. Otto and J. D. Stack, *ibid.*, p. 2328.

This is the last in a series of occasional articles on artificial intelligence. Previous articles appeared in Science, *24 February, p. 802; 23 March, p. 1279; 27 April, p. 372; and 15 June, p. 1225.*