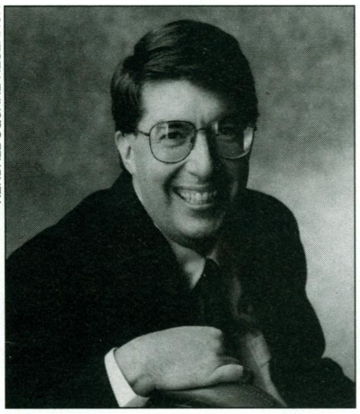
Massively Parallel Machines Usher In Next Level of Computing Power

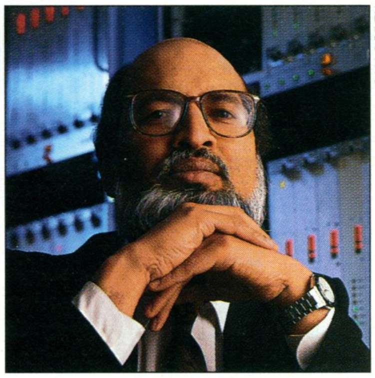
How greedy can we get? Only a few years ago, most scientists would have been grateful for access to a supercomputer performing, say, 100 million operations per second. Today, the fastest supercomputers perform 20 billion or so operations per second—and scientists still aren't satisfied. A truly reliable climate model, geophysicists say, probably needs a computer 1000 times faster than today's best. And biologists modeling protein folding or doing rational drug design can't wait to get their hands on a machine with at least a trillion operations per second. Everywhere, the story is the same: researchers in materials analysis, earthquake prediction, quantum chromodynamics, cosmology, and aircraft design all want more.

KENDALL SQUARE RESEARCH



Massively parallel visionaries: Henry Burkhardt and Arvind.

SAM OGDEN



Thank goodness for massively parallel computers. Traditional supercomputers may still be getting faster, but they're quickly approaching both fundamental limits and prices beyond the reach of all but a handful of customers. But by hooking together thousands of individual microprocessors, computer makers can create a fast, relatively inexpensive machine whose power can be increased almost indefinitely merely by adding on more processors. Such massive parallelism, some experts predict, will increase the maximum computer speed by a factor of 1000 or more in only 5 years. The same proportional increase took some 25 years for traditional computers.

Already massively parallel computers are taking over many scientific calculations from supercomputers. At Sandia National Laboratories, SDI researchers use massively parallel machines made by nCUBE of Beaverton, Oregon, and Thinking Machines Corp. of Cambridge, Massachusetts, to perform theoretical studies of intense electron beams that travel at almost the speed of light. At the Caltech supercomputing consortium, biologists working on an Intel Touchstone Delta computer calculated evolutionary trees of bacteria by cross-comparing the genetic sequences of 473 known bacteria species. And at Columbia University in New York City, theoretical physicist Norman Christ and students strung together hundreds of off-the-shelf components to build their own massively parallel machine to run calculations in quantum chromodynamics (QCD)—the theory of matter at the level of quarks and gluons.

NORMAN CHRIST, COLUMBIA UNIVERSITY

Hundreds of individual processors are hooked together to form Norman Christ's machine.

Christ, like most scientists now using massively parallel computers, works on a problem that lends itself to the parallel approach. He breaks up space-time into a lattice of points and performs calculations at each point to get a Feynman path integral—the basic computational tool in QCD. Says the Columbia physicist: "It's easy to do in parallel because all the processors can be doing the same thing." And just as important, the processors don't have to talk to each other very much: QCD is a situation of "propagating influences, where neighbor affects neighbor affects neighbor," Christ explains, and this physical simplicity creates a computational simplicity—each processor must communicate with its four nearest neighbors in the 256-node computer, but not with each of the other 251 as well.

Not all research adapts so well to today's massively parallel computers, however. Mark Pinto at AT&T Bell Laboratories, who models the performance of transistors and other semiconductor devices before they are built, prefers supercomputers with large, fast memories. "We would really have a communications bottleneck" on a massively parallel machine, he says, because a single electric charge at one point of a simulated transistor affects the computations at every other point. On a parallel computer, every processor would need to exchange information with every other one, slowing the computation down to a relative crawl.

Nonetheless, most observers say it's just a matter of time before massively parallel computers take over just about every computationally intensive application. Even Cray—the company whose name is synonymous with supercomputers—has announced it's getting into the massively parallel business. It's simply a matter of bytes for the buck.

For supercomputer makers such as Cray, NEC, and IBM, it's been getting harder and harder to squeeze out a few more of those bytes per buck. For over a decade, they have continually beefed up their machines by cramming more and faster transistors onto the state-of-the-art processors that power the computers. (A supercomputer has one, or at most a few, processors, all connected to one large memory.) But not only is each generation of these processors more expensive than the last, they are running up against physical barriers, such as the speed of light, which limits maximum signal speed.

It was this limit that led Danny Hillis in the mid-1980s to propose stringing together thousands of slower, cheaper processors to make a machine that could not only compute faster than a supercomputer but do it much more cheaply. Hillis, who started Thinking Machines to put his ideas into practice, was scoffed at and disregarded at first, but he's had the last laugh. Not only have his company and several others built massively parallel machines that rival supercomputers, they're about to leap far ahead.

The key to this breakthrough has been the amazing progress in microprocessors—the chips that power personal computers and workstations. Faster and cheaper each year, some are now as powerful as the original Cray-1 supercomputer of the mid-1970s. That's led Hillis to propose a machine with 16,000 microprocessors that could perform 2 billion operations per second—100 times as many as the fastest supercomputer today (see box).

Still, the experts say that two major obstacles must be overcome if massively parallel machines are to come into common use and not be limited to special problems like Christ's QCD calculations. One is the "memory bottleneck"—the traffic jam that arises when routing data around a machine with hundreds or thousands of individual processors, each of which might need information from any one of the others. The second is the difficulty of writing programs for massively parallel machines, an arcane art practiced now by only a few specialists.

In a massively parallel computer, it's impractical to put all the memory in one place—the different processors would get in each other's way as they all tried to retrieve data at the same time—so the memory is distributed equally among the processors. But with hundreds or thousands of individual memories, a programmer has to keep track of where each bit of data is—a Herculean task. And since data has to be swapped around to the right locations before the processors can do their work, the communication of data can end up taking 100 or 1000 times as long as the calculations themselves. Unless the processors seldom need data from other memories, this kills the advantage of massive parallelism.

"The only solution is to build a shared memory machine," says Henry Burkhardt, founder of Kendall Square Research of Waltham, Massachusetts. Kendall Square has designed what many industry observers say is the most promising solution to the memory problem—a "virtual shared memory." Although the memory is still physically located in hundreds of different locations connected to individual processors, the computer's wiring and operating system are set up so that the memory appears to be all in one place. When a processor asks for a piece of data, it doesn't worry whether that data is in its own memory or halfway across the computer.

Programming problems. Besides solving the memory problem, researchers must also simplify the programming of massively parallel machines, says MIT computer scientist Arvind. (Like Madonna or Sting in the entertainment world, Arvind uses only one name; he dropped the other before high school.) "If programming could be made dramatically easier," he says, "then we would see a dramatic increase in the applications."

Computer programmers use languages such as Fortran that are inherently sequential, Arvind notes. Even when a problem is parallel, such as adding two matrices, Fortran breaks it up into a series of calculations done one at a time. "We need languages where unnecessary sequencing is not introduced," he concludes. Which is why Arvind is developing just such a language, as part of a collaborative project between MIT and Motorola to develop both the hardware and software for a new type of massively parallel machine. The key to this or any other language for parallel machines, he argues, is to make the parallelism implicit so that the programmer doesn't have to worry about it.

Other researchers agree with Arvind that software is a problem, but many disagree with his solution. "Arvind's work ignores that a lot of programs already exist," written in such languages as Fortran, Burkhardt says, so a lot of work would go to waste if a new programming language became standard. A better idea, he contends, is to design massively parallel computers and their operating systems so that programs written in Fortran and other existing languages can be transferred easily to them. Whatever the approach, everyone agrees that software written for one massively parallel computer should work on others with different numbers of processors, and that it should be easy to add two programs together to create a third, larger program—a must if programmers are going to avoid having to write each new program from scratch.

Once the memory and programming problems are ironed out—something nearly everyone expects to happen in the next few years—massively parallel machines should revolutionize computing. "In 10 years, we could build a machine with thousands of teraflops [a million times faster than today's computers] for only a few million dollars," Burkhardt bravely predicts. Perhaps then scientists would have enough computing power. And, then again, perhaps not. Scientists are a greedy lot.

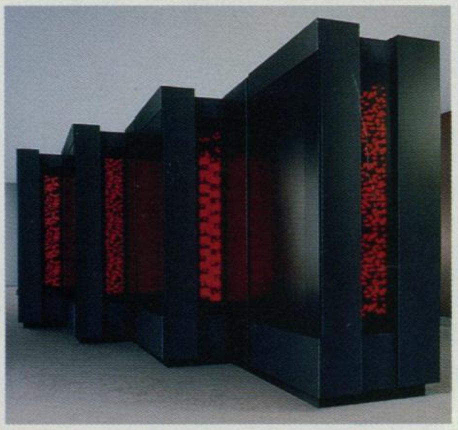
—R.P.

The Rush-to-a-Teraflop Flap

The race is on to build a teraflop computer—one that can handle a trillion operations per second. But one of the leaders in the field of computing warns that all the hype might lead to "a teraflop before its time."

The teraflop goal was made official in 1990 when D. Allan Bromley, President Bush's science adviser, looked at the then-fastest computer—around 10 billion operations per second—and asked for 1 trillion by 1996. But massively parallel computer manufacturers don't want to wait that long.

"We could build a teraflop machine today," said Danny Hillis, chief scientist of Thinking Machines Corp., at a press conference last fall. For a mere $200 million, the company's newest offering, the Connection Machine 5, could be outfitted with 16,000 processors and would run at a peak speed of two teraflops. Not to be outdone, other makers have announced their own lines that can be scaled up to teraflop size—if someone has the money.



THINKING MACHINES CORPORATION

Thinking Machines says its CM-5 could be scaled up to a teraflop, but should it?

So far, no one has anted up, but scientists aren't waiting. In January, a group of biologists held a workshop to muster interest in a massively parallel computer to help in such areas as protein structure calculations and rational drug design (see *Science*, 24 January, p. 391). Paul Bash, the Florida State University biologist who hosted the workshop, says the group would like a 1-teraflop machine, which Thinking Machines says it can make for about $100 million.

Last August, even before Hillis' public announcement, a group of 40 physicists asked the Department of Energy (DOE) to buy a teraflop machine for calculations in particle physics. Unfortunately for the scientists, their timing couldn't have been worse: In response to strict budget ceilings, DOE was looking for research programs to cut and was in no mood to add new commitments. Still, says Columbia University theorist Norman Christ, the physicists hope DOE will have a change of wallet.

But Gordon Bell, who is one of the most respected gurus in the computer field, hopes it won't (see p. 64). "It's a tremendous waste of government funds to buy a teraflop machine at this time," the former chief designer at Digital Equipment says. Why? Prices on microprocessors, thousands of which would be linked to make a teraflop computer, are dropping fast, and, says Bell, "if you wait three years, [the price of a 1-teraflop machine] will be down by a factor of four."

Furthermore, the 1-teraflop computers that are the subject of all the hype aren't really single computers at all, Bell contends, but are "multicomputers": thousands of individual microprocessor/memory units linked in loose networks much like the networks of workstations in many large laboratories. Until manufacturers offer massively parallel machines that are truly integrated and can be programmed and run as single computers, Bell has some advice: "It would be better to buy a bunch of workstations." Don't shell out $100 million just to be the first on the block with a teraflop. You could find yourself with a megapriced flop on your hands.