COMPUTER SCIENCE

'Self-Tuning' Software Adapts to Its Environment

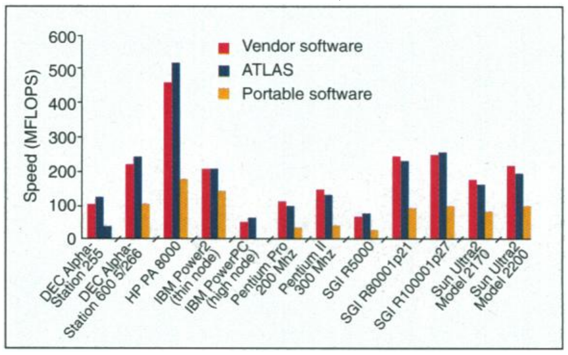
Programmers are making computers do the hard work of adjusting software to make the best use of the hardware it runs on

In computation, speed is of the essence, as children quickly learn when they start taking arithmetic tests. But advances in computer capabilities, which have tempted researchers to develop more complex programs and tackle larger problems, have made speed trickier than ever to achieve. Programs tailored to run fast on a particular machine may not work at all on others, while "portable" codes often run inefficiently on all platforms. Programmers expend a lot of effort fitting software to hardware—and the target keeps moving.

Now, some researchers are figuring out how to make the computers do the work. New "self-tuning" software probes the capabilities of the hardware it's running on and generates code that takes advantage of what it finds. The software—mostly designed for scientific computation—creates subroutines for common computational tasks, such as multiplying matrices. The gain in efficiency from such subroutines "can be dramatic," says Jack Dongarra, a computer scientist at the University of Tennessee, Knoxville, who is developing some of these programs. "I'm not talking about 10% or 20%, I'm talking about 300% improvement." So far, the self-tuning bandwagon has picked up only the most hardcore hardware users. But its proponents say that self-tuning software will soon be the only way for programmers to keep pace with their machines' capabilities.

These programs address variations in the way computers cope with a universal problem: the mismatch between today's speedy processors and the sluggish rates at which data can be gathered from memory. Computer architects deal with the discrepancy by creating hierarchies of memory—from the big, slow main memory, through several levels of smaller but faster "cache," up to the "registers," where the processors do their thing. If a computer were a supermarket, with high-volume, impulse items placed at the checkout lines (the registers), other items stocked on shelves, and yet more in the storeroom, software would be a shopping list. But unless the organization of the list matches that of the store, the unlucky consumer may dash back and forth between the same aisles and make repeated calls for help from the storeroom. The obvious solution is to reorder the list; the hard part is to find an optimal reordering.

"In the old days, people would get a new machine, run some experiments by hand, and make some changes to the structure of the software to get it to run very efficiently," explains Dongarra. "What's being done in this new paradigm is to put that [kind of] smarts into a program." He and colleagues R. Clint Whaley and Antoine Petitet have developed such a program—dubbed ATLAS, for Automatically Tuned Linear Algebra Software—to create efficient algorithms for multiplying matrices, the rectangular arrays of numbers that underlie virtually all scientific computation. Similar software, called PHiPAC (Portable High-Performance ANSI C), is being developed by Jim Demmel and colleagues at the University of California, Berkeley.



Performance-tuned. Graph shows that ATLAS generates algorithms for a linear-algebra operation that run as fast as vendor-provided software on many different processors.

An efficient matrix-multiplication program breaks each matrix into smaller blocks, multiplies the pieces, and then recombines the results. The trick is to decide, for example, how big the blocks should be and in what order to multiply them—and that depends on the computer's own memory structure. ATLAS and PHiPAC explore the computer's cache structure and match it to a set of parameters that describe a range of possible algorithms. They then settle on an algorithm that minimizes the movement of data up and down the memory hierarchy.

The tuning process is slow: ATLAS takes a couple of hours to run, and PHiPAC typically takes several days. But once the self-tuning software finds a good algorithm, the result can be used for the lifetime of the machine in any program that requires matrix multiplication.

Another new self-tuning program, whimsically called the Fastest Fourier Transform in the West—the FFTW, for short—is virtually cost-free. It takes mere seconds to adapt itself to a given machine, then cranks out computations at a rate that rivals implementations painstakingly written for a particular computer. The program is the brainchild of two Massachusetts Institute of Technology graduate students: computer scientist Matteo Frigo, who has since earned his doctorate, and physicist Steven Johnson.

The FFTW creates subroutines for computing Fourier transforms, an operation that is crucial in all kinds of scientific computation. A Fourier transform finds the periodic patterns in seemingly jumbled data, such as the waxing and waning of ice ages as recorded by geological formations.

The basic procedure for computing Fourier transforms quickly, known as the Fast Fourier Transform, or FFT, was introduced in the 1960s. It works by decomposing a single transform, which analyzes a large data set, into smaller ones, each of which can be broken down still further. Small transforms are much easier to compute than large ones are, and there's a net saving even after all the small computations are counted together.

"The FFT algorithm actually gives you a lot of freedom in how you decompose the transform," Johnson says. A transform of size 100, for example, can be broken into two transforms of size 50, or four of size 25, and so forth. The challenge is to find a decomposition that fits neatly into the computer's memory hierarchy. The tuning phase of the FFTW solves a test problem, trying all possible decompositions until it finds the fastest one for a transform of a specified size. That's a more limited problem than the matrix programs face, which is why the FFTW is so speedy. Frigo and Johnson say they are getting thousands of hits daily on their FFTW Web site (www.fftw.org), from clients who range from astrophysicists wanting to analyze pulsar data to "people who want to tune their guitar by computer," Johnson says.

Computer vendors, who often provide libraries of code designed to run quickly on their own machines, are also keen on the new self-tuning software. "These are great tools," says Bruce Greer, a senior software architect at Intel. He says Intel programmers "will use FFTW as a benchmark against which to judge the quality of our software," before tweaking their code to squeeze out additional performance. "If we can't beat FFTW, then we probably haven't tried hard enough," he says.

–BARRY CIPRA

CREDIT: WHALEY AND DONGARRA