Editor: Rubin Landau, rubin@physics.oregonstate.edu

Multicore Processors for Science and Engineering

By Pam Frost Gorder

There's no question that multicore processors have gone mainstream. These computer chips, which have more than one CPU, first hit the consumer market less than two years ago. Today, practically every new computer has a dual-core (two-CPU) chip, and Intel just launched a quad-core chip with four CPUs. One of 2006's most in-demand holiday gifts was Sony's PlayStation 3, which boasts a "cell" chip with nine CPUs for faster and more realistic video gaming.

Multicore systems might offer advantages to gamers, but what about researchers? David A. Bader, who directs a new research center at Georgia Tech devoted to cell technology, says that making the most of multicore systems will require new tools, new algorithms, and a new way of looking at programming.

Embrace Concurrency

"We've known for some time that Moore's law was ending, and we would no longer be able to keep improving performance," Bader says. "The steady progression from symmetric multiprocessing to putting many functional units on a chip to multicore has been a long time coming." Software ran faster year after year, not because of software innovations, but because chip makers kept adding transistors to the standard single-processor architecture. Now, he says, clock speeds are capped out at around 4 GHz: "If we want faster speeds, we have to embrace concurrency and make use of multiple processors on the chip at once."

Bader heads the Sony-Toshiba-IBM (STI) Center of Competence at Georgia Tech, where researchers will develop applications for the Cell Broadband Engine (Cell BE) microprocessor—the chip that powers the PlayStation 3, as well as IBM's QS20 blade servers. The Cell BE is already being developed for aerospace, defense, and medical imaging; the new research center will focus on scientific computing and bioinformatics. Bader also received a Microsoft research grant to develop algorithms that exploit multicore processors. He'll adapt a library package called Swarm (SoftWare and Algorithms for Running on Multicore; http://sourceforge.net/projects/multicore-swarm/) that he began building in 1994.

Although a huge industry push toward multicore systems exists today, there wasn't one when Bader first began working on Swarm. Another computer scientist who started thinking about multicore even earlier—albeit, on the hardware side of things—is Stanford University's Kunle Olukotun. He recalls that when he and his colleagues started talking about multicore architectures in the early 1990s, they received a cool reception. "Back then, people thought that single-core processors still had a lot of life in them," he says. But by 2001, he was working with Sun Microsystems to commercialize his first multicore chip, the Niagara. They designed it to work 10 times faster than existing devices with half the power consumption.

In the end, what drove the industry to multicore technology wasn't just the need for more processing speed, Olukotun says—it was the need for less heat and more energy efficiency. Less heat because the fastest chips were heating up faster than the average fan could cool them down, and more energy efficiency because single-core chips rely on tightly packed power-hungry transistors to get the job done.

Compared to several single-core chips, a multicore chip is easier to cool because the CPUs are simpler and use fewer transistors. This means they use less power and dissipate less heat overall. As for performance, each multicore CPU can work on a different task at the same time. Parallel processing used to require more than one chip or clever algorithms to simulate parallel processing from the software side. In a multicore processor, however, parallelism is already built in.

Modeling (and More) on Multicore

So what will multicore processors mean for researchers? Bader says the built-in parallelization should seem natural to people who model computationally intensive problems because they're already accustomed to using parallelized codes on computer clusters or supercomputers. But the average scientist will need tools. "[Scientists] may use Matlab or Mathematica or other standard packages, and they're going to have to rely on those frameworks to make use of parallelism. Others who are developing scientific codes are

Copublished by the IEEE CS and the AIP 1521-9615/07/$25.00 © 2007 IEEE

Parallelization Tools

Computer scientists are working on tools to help parallelize the sequential algorithms used in research today. Each of these projects recently received funding from the US National Science Foundation to speed their development:

- Most large-scale scientific programs have irregular data structures, such as tree and graph structures, that are hard to parallelize. Keshav Pingali, the W.A. "Tex" Moncrief Chair of Grid and Distributed Computing at the University of Texas at Austin, heads the Galois project, which aims to develop a compiler that can parallelize irregular algorithms for multicore processors. Galois uses "optimistic parallelization" to address what happens when more than one core accesses the same data at the same time. It runs a program in parallel, but if the data becomes corrupted, it aborts the computations and re-runs them sequentially.
- Eric Polizzi, assistant professor of electrical engineering and computer engineering at the University of Massachusetts Amherst, is developing a parallel linear system solver called SPIKE, and integrating it into a nanoelectronics simulator, where it can help with quantum calculations (www.ecs.umass.edu/ece/polizzi/Softwares.html). He plans to create a toolkit that can automatically provide nonexpert users with the most efficient version of SPIKE for their hardware schemes.
- Rastislav Bodik, assistant professor of computer science at the University of California, Berkeley, is developing a different tool to make parallelization easier. Scientists start with their own sequential code and then "sketch" an outline of what they'd like the implementation to be, and a compiler does the rest (www.cs.berkeley.edu/~bodik/research/sketching.html). "We had scientific programmers in mind when we designed the sketching language," Bodik says.

going to have to think differently about those problems and ways of revealing concurrency," he says.

The problem is that most programmers—truly, most humans—think sequentially, so most codes are written to run sequentially. Parallelizing them can require heroic effort, Bader says: "We could rely on compilers to convert the code that we already have. But except for a few situations where data has a very simple structure, we haven't seen compilers as a magic bullet to get performance."

When Olukotun and his colleagues designed Niagara, they optimized it to run commercial server applications that were already highly multithreaded, meaning they split tasks into "threads" of execution—instruction sequences that run in parallel. Now he's working on a new technique called

Web Trends

For a brief look at current events, including program announcements and news items related to science and engineering, check out the following Web sites:

- Advanced Technological Education (ATE; www.nsf.gov/publications/pub_summ.jsp?ods_key=nsf07530). This program emphasizes two-year colleges and their role in providing technicians to high-technology fields. Deadline for proposals is 11 October 2007.
- Canada Creates National High-Performance Computing Network (www.innovation.ca/media/index.cfm?websiteid=482). Using C$88 million dollars from the Canada Foundation for Innovation (CFI) and the Natural Sciences and Engineering Research Council of Canada (NSERC), Canada will create its first national high-performance computing (HPC) network that unifies all HPC efforts across the country. All seven of Canada's HPC Consortia will participate in the project.
- Cornell Awarded Grant to Lead NSF's Broadening Participation in Computing Program (www.tc.cornell.edu/News/2006/061214.htm). A new grant will fund the Worlds for Information Technology and Science (WITS) project, which will look for ways to attract women and minorities to computing with service learning.
- Industry/University Cooperative Research Centers Program (I/UCRC; www.nsf.gov/publications/pub_summ.jsp?ods_key=nsf07537). The I/UCRC program's goal is to foster partnerships among industry, academia, and government. Letters of intent are due 29 June 2007.
- Mathematical Sciences: Innovations at the Interface with Computer Sciences (MSPA-MCS; www.nsf.gov/publications/pub_summ.jsp?ods_key=nsf07534). The MSPA-MCS program fosters collaboration between mathematicians, statisticians, engineers, and scientists. Proposal deadline is 12 March 2007.
- Integrative Graduate Education and Research Traineeship Program (IGERT; www.nsf.gov/pubs/2007/nsf07540/nsf07540.htm). The US National Science Foundation (NSF) is soliciting proposals for the IGERT program, which was developed to help meet the educational needs of US PhD students in the science and engineering fields pursuing research and education careers. Preliminary proposal deadline is 5 April 2007.
- National Science, Technology, Engineering, and Mathematics Education Digital Library (NSDL; www.nsf.gov/pubs/2007/nsf07538/nsf07538.htm). The NSF is accepting proposals for its NSDL program, which seeks to create an online network and national library for science, technology, engineering, and mathematics education. Proposal deadline is 11 April 2007.

thread-level speculation that lets users parallelize sequential algorithms automatically for multicore architectures.

"The idea with speculation is that the parallelization may not always work, and you can detect at runtime whether it's working. When there is no parallelism in the application, you still get the same result as if you had a single-core processor. You can think of it as a safety net." He sees colleagues in the sciences using more dynamic algorithms that are difficult to parallelize. "Be it seismic analysis for oil exploration or molecular dynamics for protein folding or probabilistic inference, these types of algorithms could really take advantage of the speculation," Olukotun says.

Petaflops by 2010

Multicore technology is taking hold in supercomputing, where the goal is to reach petaflop (one quadrillion calculations per second) capability by the end of the decade. In 2006, the makers of the Top500 Supercomputer Sites list (www.top500.org) reported that 100 of its denizens now use dual-core chips, and that this number is expected to grow. At the 2006 International Supercomputing Conference (ISC) in Dresden, Germany, multicore computing was called one of the year's advances—and one of two technologies that would guide supercomputing to the petaflop goal. As Thomas Sterling, professor of computer science at Caltech, wrote in his ISC review, 2006 marked a turning point in the quest for such machines (www.hpcwire.com/hpc/709078.html).

In particular, Sterling pointed to the multicore systems in the Top500 list, including the top-ranked IBM BlueGene/L system, which achieved 280.6 Tflops (trillions of calculations per second). "While the majority of such systems are dual-core," he wrote, "next-generation systems are rapidly moving to quad-core. And it is expected that this trend will continue with Moore's law over several iterations. However, it is recognized that the shift to multicore brings with it its own challenges. […] Even for the world of supercomputing, this trend to multicore will impose a demand for increasing parallelism. If, as is expected, this trend continues, then the amount of parallelism required of user applications may easily increase by two orders of magnitude over the next decade."

Scientists who already run large simulations or process massive amounts of data in parallel can look forward to some improvements from multicore systems. "Big science" problems in Earth science, atmospheric science, and molecular biology are among those that would benefit.

Jim Gray, manager of the Microsoft Research eScience Group, works with the Sloan Digital Sky Survey (SDSS), which holds the world's largest astronomical database (approximately 3 Tbytes of catalog data and 40 Tbytes of raw data). The SkyServer (http://skyserver.sdss.org) lets as-

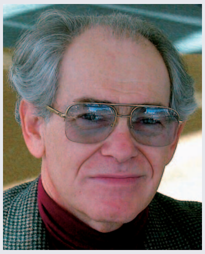
EDITORIAL OFFICE

COMPUTING in SCIENCE & ENGINEERING
10662 Los Vaqueros Circle, Los Alamitos, CA 90720 USA
phone +1 714 821 8380; fax +1 714 821 4010; www.computer.org/cise/

IEEE Antennas & Propagation Society

IEEE Signal Processing Society

Observatoire Landau



Ohio Creates a Statewide Virtual School of Computational Science

By Rubin Landau, news editor, with help from Steve Gordon, Ohio Supercomputer Center

Previous pioneers braved the Oregon Trail (http://en.wikipedia.org/wiki/Oregon_Trail) in wagons pulled by oxen and built structures with logs and sod. Similarly, modern pioneers venture into the established educational infrastructure and build virtual schools with ideas and technology.

In December 2005, the Ohio Supercomputer Center, following in the path of the Maryland Virtual High School of Science and Mathematics, created a virtual school of computational science, known as the Ralph Regula School of Computational Science. (Regula is a US Representative from Ohio, who also happens to be chairman of the House Appropriations Subcommittee charged with supporting and overseeing programs in the US Department of Education.) The school focuses on teaching computer modeling and simulation to directly address the problem of the US falling behind in computational science and in teaching science. The virtual school isn't the effort of a single individual who has seen the light and is working from the bottom up—it looks like a grand enough effort to initiate a systemic change in K–20 education, so it's worth spreading the news about—it's a collaboration among the Ohio Board of Regents, the Ohio Supercomputer Center, the Ohio Learning Network, Ohio's colleges and universities, and industries.

tronomers and educators access the catalog database. He says that, so far, processor speed isn't as important to SkyServer data access as the speed of the disks that store the data. The project generally has more CPUs available than it needs. Still, Gray says, "The SDSS image processing pipeline is 10,000 instructions per byte. That used to be a room full of machines, but the 4-GHz processors with four cores will run at more than 1 Mbyte per second, so we only need a few multicore machines to process the data."

Johns Hopkins University research scientist Ani Thakar adds that he and his SDSS colleagues are working hard to keep their CPUs busier, in part by parallelizing data access and "bringing the analysis to the data as far as possible, rather than the other way around" to minimize disk input/output. "I think in the near future, our fraction of CPU usage will steadily increase and we will be able to benefit considerably from the multicore design," Thakar says.

As I preach (rant?) in my proselytizing talks on computational science and computational physics, American students and faculty members often appear to believe that because they're surrounded by high technology in the land where much of it is developed, they'll somehow naturally be the industry's future leaders. Yet, I have found that students in developing countries, possibly realizing that they don't have the industrial might and capital of the US, have a more advanced view about using modern computations and the newly developing scientific grids and distributed databases to do innovative and forefront work. Not only do they work harder to learn the basics better than our students, they're keen to apply their education to something significant.

I believe it will take an effort like the one in Ohio to make a significant change in attitudes about how to do science and engineering in the future. Specifically, the Ohio virtual school will focus students on solving real-world problems by applying scientific principles and analyzing data. The virtual school's initial steps have focused on making undergraduate computational science instruction available statewide, regardless of whether campuses have the resources to offer their own courses. They're developing an undergraduate computational minor, slated for statewide introduction this year, that will let science and engineering majors achieve competencies in a number of areas. The minor includes

- simulation and modeling;
- programming and algorithms;
- differential equations and discrete dynamical systems;
- numerical methods;
- optimization;
- parallel programming;
- scientific visualization;
- one discipline-specific course; and
- capstone research or internship experience.

Ten institutions are developing the computational science materials and will share the courses and instruction. The actual degrees won't be offered by the virtual school, but by the participating colleges and universities.

In addition to a computational minor, the virtual school is preparing a computational science certificate program aimed at people presently in the workforce. The focus group ranges from displaced workers seeking new skills in an emerging field to active scientists and engineers wishing to upgrade their knowledge of computational approaches to solving problems. The certificate program will be standardized and coordinated with industry and initially focus on basic simulation and modeling, engineering and design, and biochemistry. At a lower level, the virtual school has received approval to develop a computational course that will be used in Ohio high schools. Teachers will be trained on the use of the materials in a workshop during the summer of 2007, and begin offering the course to their students in the fall of 2007. The high-school students will then be in a pipeline in which they can receive additional computational science and engineering education as they move through Ohio's higher education system.

Will this work? It's hard to predict, but at least it indicates that computational science is being institutionalized statewide as a way of improving both education and the workforce. This no doubt requires the long-term commitment of people in state government, university administration, and teaching, as well as at the Ohio Supercomputer Center. I encourage our readers to also support them—although they missed being able to call themselves the nation's best in college football this year, I for one will cheer if the state of Ohio is number one in computational science education.

Thinking in Parallel

For decades, programmers have been trained to write sequential algorithms. To Bader, the ability to write parallel code is a different kind of skill, one that has nothing to do with a programmer's intelligence, but rather his or her ability to think broadly. "I'm convinced that it's an art," he says. "You either get it, or you don't." He's training students at Georgia Tech to think in parallel—and to think of how their programs connect to larger issues in science and engineering.

"I think this is a really exciting time—the first time in 20 years that we've seen really disruptive technologies in computing," Bader says. "Multicore is a disruptive technology—and I mean that in a good way—because it's only when you have disruption of the status quo that new innovations can impact technology with revolutionary advances."

Universities that embrace this philosophy could reap an added benefit. Bader says Georgia Tech has seen a boost in computer science enrollment; nationally, the number of students interested in the major is falling. "Computer science has in some sense become stagnant because many students today don't see how computer science impacts the world," he says. Georgia Tech has reorganized its computer science program to create a computational science and engineering division to tie programming to the idea of solving real-world problems. So have the University of California, Berkeley, and the University of Texas at Austin, and Bader predicts that more universities nationwide will soon follow. Multicore computing is helping to kick-start the change.

Pam Frost Gorder is a freelance science writer based in Columbus, Ohio.