Designing exascale computers

July 23 2014, by Brian Hayes

Credit: Michelle Borkin, Kaxiras Lab

"Imagine a heart surgeon operating to repair a blocked coronary artery. Someday soon, the surgeon might run a detailed computer simulation of blood flowing through the patient's arteries, showing how millions of red blood cells jostle and tumble through the small vessels. The simulation would identify the best repair strategy.

With a fast enough computer, it could all be done in a few minutes, while the operation is under way."

This is the vision of Efthimios Kaxiras, the John Hasbrouck Van Vleck Professor of Pure and Applied Physics. As leader of an interdisciplinary research group, he has built just such a simulation of "hemodynamics," tracing the movements of several million blood cells through the filigree network of coronary arteries that supply oxygen to the heart muscle. But simulating just one second of blood flow—the duration of a single human heartbeat—took five hours on one of the world's fastest supercomputers a few years ago. If such models are to transform medical practice, they will need computers that run a thousand times faster.

Vastly increased computer power could also enhance the detail and verisimilitude of the models. If the simulation could reach down to the molecular level, Kaxiras says, it might directly model the way lipid molecules are deposited on the arterial wall, predicting blockages before they form.

Boosting computer performance by a factor of a thousand—and finding ways to make that power readily accessible to the scientific community—is a formidable research challenge. Within the School of Engineering and Applied Sciences (SEAS), several groups are contributing to the basic science and engineering that will be needed to create both the hardware and the software for the next generation of supercomputers. Others at SEAS and elsewhere at Harvard are poised to take advantage of the new computing machinery when it becomes available, applying it in fields such as climatology, materials science, molecular biology, and astrophysics.

The prospect of a computation-rich future also has a bearing on the educational mission of the university. For example, the master's degree programs, courses, and graduate minor recently launched by the Institute for Applied Computational Science (IACS) at SEAS focus on applying computation to cross-disciplinary challenges. In applied computation

courses, students experiment with a toolkit of traditional and new methods in applied mathematics and computer science to explore whole systems and worlds, from the nanoscale to the global, from the human genome to atmospheric chemistry and economic systems.

So-called exascale computers might allow the power of these methods to be fully realized.

Harvard's first large-scale digital computer, which came to be known as the Mark I, was conceived by Howard H. Aiken (A.M. '37, Ph.D. '39) and built by IBM. Fifty-one feet long, it was installed in the basement of what is now Lyman Laboratory in 1944, and later moved to a new building called the Aiken Computation Laboratory, where a generation of computing pioneers were educated and where the Maxwell Dworkin building now stands. (Part of the mechanism remains on exhibit in the Science Center.)

The Mark I performed additions and subtractions at a rate of about three per second; multiplication and division took considerably longer. This benchmark was soon surpassed by computers that could do thousands of arithmetic operations per second, then millions and billions. By the late 1990s a few machines were reaching a trillion (10^{12}) operations per second; these were called terascale computers, as tera is the Système International prefix for 10^{12}. The next landmark—and the current state of the art—is the petascale computer, capable of 10^{15} operations per second. In 2010, Kaxiras' blood flow simulation ran on a petascale computer called Blue Gene/P in Jülich, Germany, which at the time held fifth place on the Top 500 list of supercomputers.

The new goal is an exascale machine, performing at least 10^{18} operations per second. This is a number so immense it challenges the imagination. Stacks of pennies reaching to the moon are not much help in expressing its magnitude—there would be millions of them. If an exascale computer

counted off the age of the universe in units of a billionth of a second, the task would take a little more than 10 seconds.

And what comes after exascale? We can look forward to zettascale (10^{21}) and yottascale (10^{24}); then we run out of prefixes.

The engine driving these amazing gains in computer performance is the ability of manufacturers to continually shrink the dimensions of transistors and other microelectronic devices, thereby cramming more of them onto a single chip. (The number of transistors per chip is in the billions now.) Until about 10 years ago, making transistors smaller also made them faster, allowing a speedup in the master clock, the metronome-like signal that sets the tempo for all operations in a digital computer. Between 1980 and 2005, clock rates increased by a factor of 1,000, from a few megahertz to a few gigahertz. But the era of ever-increasing clock rates has ended.

The speed limit for modern computers is now set by power consumption. If all other factors are held constant, the electricity needed to run a processor chip goes up as the cube of the clock rate: doubling the speed brings an eightfold increase in power demand. SEAS Dean Cherry A. Murray, the John A. and Elizabeth S. Armstrong Professor of Engineering and Applied Sciences and Professor of Physics, points out that high-performance chips are already at or above the 100-watt level. "Go much beyond that," she says, "and they would melt."

If the chipmakers cannot build faster transistors, however, they can still make them smaller and thus squeeze more onto each chip. Since 2005 the main strategy for boosting performance has been to gang together multiple processor "cores" on each chip. The clock rate remains roughly constant, but the total number of operations per second increases if the separate cores can be put to work simultaneously on different parts of the same task. Large systems are assembled from vast numbers of these

multicore processors.

When the Kaxiras group's blood flow study ran on the Blue Gene/P at Jülich, the machine had almost 300,000 cores. The world's largest and fastest computer, as of June 2014, is the Tianhe-2 in Guangzhou, China, with more than 3 million cores. An exascale machine may have hundreds of millions of cores, or possibly as many as a billion.

"Big" computers have come a long way since the completion of the Mark I (left) at Harvard in 1944. Capable of approximately three calculations per second, the Mark I was "the first operating machine that could execute long computations automatically," according to IBM. Today, the fastest supercomputer is the Tianhe-2 (right), developed by China's National University of Defense Technology. The Tianhe-2 has achieved almost 34 petaflops, or 34,000,000,000,000,000 floating-point operations per second. Credit: Harvard University Archives and Jack Dongarra

In principle, an exascale computer could be built today, using chips like those installed in the Tianhe-2—just many more of them. But such a

monstrous machine would come with a monstrous electric bill: The power demand would be about 500 megawatts, half the output of a nuclear power plant. To bring exascale computing into the realm of practicality, energy consumption will have to be reduced by an order of magnitude.

Maximizing computational power while minimizing electrical consumption is one of the principal research aims of David M. Brooks, Haley Family Professor of Computer Science at SEAS. A first step in this direction is knowing where the energy goes. Which parts of a processor chip consume the most energy during various phases of operation? To help answer this question Brooks and his colleagues have created a series of measurement and simulation tools that document the performance of existing chips and predict the behavior of hypothetical alternatives. In this way they can explore the space of possible processor architectures, asking how design choices affect the energy budget. For example, is it more effective to have many small and simple cores on each chip or a few larger ones? Or some of each?

Brooks' findings emphasize the need to optimize many design parameters in concert, rather than tuning each factor individually. A common strategy for reducing the power consumption of digital circuits is to lower the operating voltage; but lower voltages also make the circuitry more sensitive to variations between one transistor and another, which can lead to errors. Furthermore, a surge of power demand in one part of a processor can cause a voltage dip elsewhere, much as the room lights dim momentarily when the refrigerator comes on. Again, the result of such a "voltage emergency" could be an incorrect computation. The traditional way of avoiding such errors is to maintain a margin of safety, so that the weakest transistor on the chip gets adequate voltage even under the worst-case operating conditions. That strategy may no longer be viable as demands for power efficiency become more stringent. Brooks and his group have been examining an alternative scheme in

which errors caused by voltage dips are caught and corrected, while a layer of management software works to minimize the number of voltage emergencies.

Power demands may also be reduced by improvements to the design of individual transistors and the materials from which they are fabricated. Sadasivan Shankar, Program Leader for Materials Design at the Intel Corporation who spent the fall 2013 semester as a Distinguished Scientist in Residence at IACS, emphasized this point in a talk at SEAS in January 2013. Shankar argues that it's no longer accurate to speak of silicon integrated circuits. Although silicon remains the substrate on which the circuits are inscribed, dozens of other elements from all regions of the periodic table are now essential constituents. New materials are needed to maintain the speed and reduce the power consumption of computers (and paradoxically, major computing resources are needed to explore the vast space of possible material compositions).

Processor chips are not the only components for which the tradeoff between speed and power consumption is problematic. The energy efficiency of memory chips has not kept up with that of processors; as a result, exascale machines will probably have to get along with less memory per processor core, or new memory devices will need to be invented. The energy cost of moving data from place to place within a computer is also claiming a larger share of the overall energy budget. These factors call for revisions of some common-sense notions about how to compute efficiently. Bob Adolf, a graduate student in Brooks' group, has spent 10 years building and working with supercomputers at the Pacific Northwest National Laboratory. "It was once standard practice," he says, "to avoid recomputing results whenever possible. But now storing or transmitting data can be more expensive than computing a value again, and so the priorities have shifted."

The constraint on interprocessor communication comes straight from the laws of physics. Grace Murray Hopper—one of the computing pioneers to emerge from Aiken's Computation Laboratory—used to carry around a one-foot-long copper wire to illustrate how far data can go in a nanosecond at the speed of light. A nanosecond is roughly the time it takes to execute a single instruction in a modern processor, so moving data more than a few feet will inevitably introduce delays.

Supercomputing took a turn in a new direction in 2008 with a machine called Roadrunner, built by IBM and installed at Los Alamos National Laboratory. In addition to 6,480 conventional processor chips, it had 12,960 specialized "coprocessor" chips, which had an unusual provenance. They were originally designed for the Sony Playstation game system, where they were well adapted to certain fairly simple but highly repetitive tasks such as displaying graphics or decompressing video. The coprocessor chips performed those tasks with remarkable speed and energy efficiency, and the aim of Roadrunner was to apply them to algorithms in scientific computing that have a similar repetitive structure.

Cherry Murray observes that there was initially some skepticism about the Roadrunner design: "People worried that all their software would have to be rewritten to work with the new hybrid architecture, including operating systems, libraries of utility programs, and software tools such as language compilers." Indeed it was a daunting task, but the conversion succeeded, and Roadrunner became the first petascale computer—the first to achieve 10^{15} operations per second.

Several more large-scale hybrid systems have been built since Roadrunner, mostly with graphics processing units (GPUs) as the coprocessors. GPUs, which evolved from the display-controller chips of desktop PCs, have hundreds or thousands of comparatively simple processing cores, all running in parallel. Hybrid computers that include

coprocessors of one kind or another now account for about a third of all the computing capacity on the Top 500 list of supercomputers.

Murray sees this development in the context of a decades-long process of evolution. In the 1980s, supercomputers relied on highly specialized, custom-built hardware, based on technologies quite different from those of mass-produced computers. In the 1990s, the high-performance community shifted to "vanilla" off-the-shelf components when they were found to have a better price-to-performance ratio. Now the trend is back to more specialized technologies again. Murray expects that heterogeneous architectures will rule scientific computing in the coming years. She cites an example: If you try to model the interactions of an electromagnetic field with solid-state matter, the two parts of the problem differ in character. The electromagnetic field extends continuously throughout space, whereas the solid has the discrete symmetries of a crystal. It would be surprising if one computer architecture were optimal for representing both systems; future computers may well have specialized subsystems tuned for different classes of algorithms.

The ultimate in specialization is a computer system designed to solve just one kind of problem. A prominent example is Anton, a fleet of computers designed and built by D. E. Shaw Research for studies of the flexing, twisting, and wiggling motions of proteins and other biological macromolecules. David E. Shaw, the company's founder, spoke about Anton at SEAS in 2013. The machine is a computing engine for the molecular dynamics method, which traces the movements of all the atoms in a molecule using a two-step procedure. Given an initial configuration of the molecule, the first step is to calculate the forces between all pairs of atoms; the second step moves the atoms "a little bit" in response to the forces. Then it's back to the first step to recalculate the forces based on the new positions, followed by a further small movement of each atom, and so on. To capture the fastest molecular vibrations, the

entire computation has to be repeated at intervals of 1 femtosecond, or 10–15 second. Before Anton, no one had simulated more than 10 microseconds of motion for large molecules of biological significance. With Anton it becomes practical to carry out 10^{12} iterations of the molecular dynamics algorithm, which amounts to a millisecond of biological time—100 times the previous record, and long enough to observe key events such as protein folding and enzymatic action.

Internally, Anton relies on a high level of parallelism and also on custom integrated circuits designed for this specific application. Shaw estimates it is "100 times faster than the fastest general-purpose supercomputers—but only for molecular dynamics simulations."

When you upgrade to a new laptop computer, much of your old software will probably work on the new model, and perhaps run faster. In the world of supercomputing, transitions from one generation to the next are not so effortless. The migration is not just a matter of reinstalling software, or even recompiling it from the source code. Getting the full benefit of a new architecture almost always requires adjustments at the algorithmic level.

SEAS Dean Cherry A. Murray peruses the Mark I operating handbook with Martin Gordon '57 (left) and Hugh Blair-Smith '57 (right). Gordon and Smith were Harvard roommates, back in the day, and Gordon was one of the last people to program the Mark I before the machine was retired in 1959. Credit: Eliza Grinnell

Kaxiras' blood-flow simulation provides an example. The usual approach to fluid dynamics calculations is numerical solution of differential equations, but the Kaxiras group adopted a different scheme, called the lattice Boltzmann method, in which continuous flow is approximated by the movement of "fictitious particles" through a gridlike three-dimensional lattice. This method was chosen in large part because the computation is readily partitioned among thousands of processors. Events at any one node of the lattice depend only on nearby nodes, reducing the need for data movement. Nevertheless, a recent reexamination of the algorithm showed that communications bandwidth is what limits performance; the full computational capacity cannot be put

to use because processors stand idle while waiting for data to be fetched from memory or transmitted from other processors. In adapting the Blue Gene/P code to run on the larger Blue Gene/Q in 2012, overall speed was improved by further limiting the data traffic between nearby nodes of the lattice, even though that means some values have to be recomputed.

Amanda Peters Randles, Ph.D. '13, had primary responsibility for adapting the software to run on the Blue Gene series of supercomputers. (Earlier in her career she had worked on the IBM Blue Gene team; she is now a Lawrence Fellow at Lawrence Livermore National Laboratory.) She reports that getting the simulation to run efficiently on 300,000 cores took a year and a half, and since then the process of updating and revising has never stopped. Recent changes introduced a different set of software libraries, and she rewrote the entire program in a different programming language (C rather than Fortran). Keeping up with changes in hardware can be "a little frustrating," she says, but "it's also an interesting problem-solving challenge."

Hanspeter Pfister, An Wang Professor of Computer Science and director of IACS, believes the move to exascale will call for fundamentally new programming models. "We're about to hit the wall," he says. Even at petascale, only the benchmark programs used to rate and rank the machines run at full speed; other software may reach only 10 percent. Other elements of the software infrastructure—operating systems, file systems, and the "middleware" for connecting to databases and networks—are at their limit.

Many programs now running on highly parallel computers are built on the Message-Passing Interface, or MPI. As the name suggests, this collection of utility programs and protocols allows parallel programs to communicate and synchronize their activities by passing messages. At a lower level, computations within GPU chips are most often managed

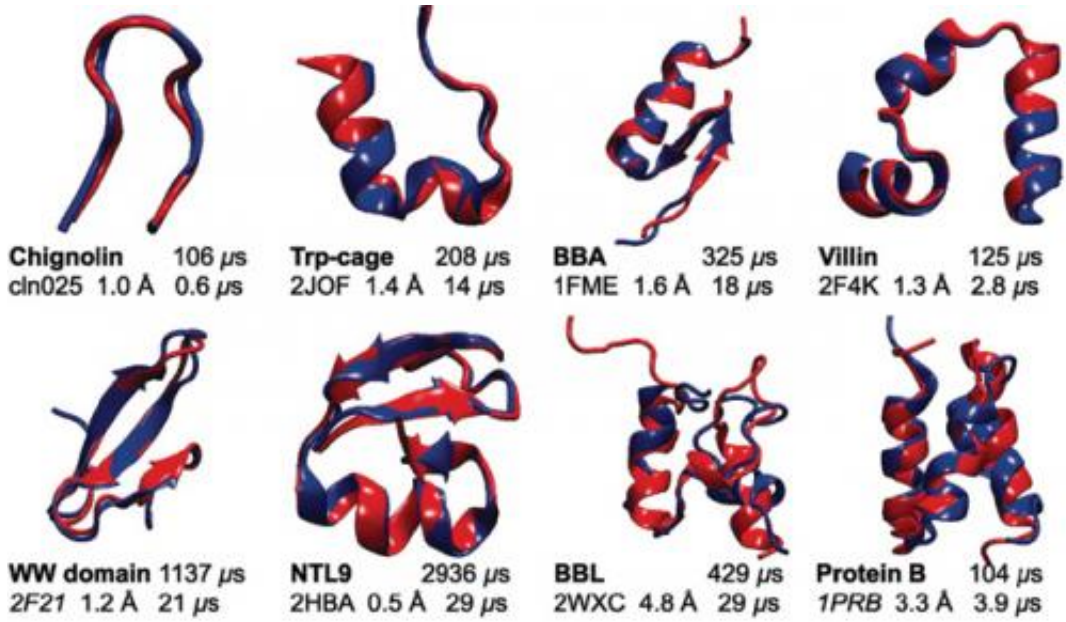
with a system called CUDA (Compute Unified Device Architecture), which allocates tasks to the thousands of cores and provides analogous facilities for communication and synchronization. These tools will not scale to the next level, Pfister says. "We can't be thinking about a billion cores in CUDA. And when the next protocol emerges, I know in my heart it's not going to be MPI. We're beyond the human capacity for allocating and optimizing resources."

The remedy, Pfister suggests, may be "brute force." Some of the labor of managing concurrent computation should be moved into hardware, which already coordinates multiple "threads" of execution in multicore processors. And some new level of abstraction is needed to relieve the programmer of responsibility for micromanaging parallel processes. One possible model is the MapReduce protocol adopted by Google for computations on massive data sets. Under MapReduce, tasks are automatically parceled out to processors, then results are gathered up, all without the need to specify where and when each piece of work is done. Existing implementations of MapReduce are not ideal for scientific computing tasks, but the basic principle might be adapted to the needs of science just as GPUs have been. "CUDA meets MapReduce," Pfister suggests.

The HELIX project tackles the problem of parallel programming from another direction. HELIX is a programming-language compiler that automatically arranges for independent sections of a program to run simultaneously. HELIX is the creation of Simone Campanoni, a postdoctoral fellow at SEAS, who works with Brooks and with Gu-Yeon Wei, Gordon McKay Professor of Electrical Engineering and Computer Science.

Still another tool for parallel computing comes from Leslie G. Valiant, the T. Jefferson Coolidge Professor of Computer Science and Applied Mathematics. Valiant, a Turing Award winner, has created a "bridging

model" that might help close the gap between parallel hardware and software. In sequential computing (that is, with one processor), programmers can generally ignore the idiosyncrasies of a particular machine and think in terms of an idealized or abstract computer, called the von Neumann machine. Valiant proposes an analogous model for parallel computers that allows programmers to ignore details such as the pattern of connections among the processors, and to write programs that are portable among different machines. Companies such as Google and Facebook have recently adopted Valiant's bulk synchronous model for their most onerous computing tasks.



David E. Shaw and his collaborators used a supercomputer called Anton to model at high resolution the molecular dynamics of protein folding, a very rapid process in which polypeptide chains spontaneously self-assemble into characteristic 3D structures. Here, each experimentally derived protein structure is shown in red, and the results of each simulation in blue. Credit: Kresten Lindorff-Larsen, Stefano Piana, Ron O. Dror, & David E. Shaw [2011] and Science/AAAS

Hardware and software each have an important role to play in the development of the exascale era. Many believe that "co-design"—bringing hardware engineers together with scientists and programmers—will result in the best match between what users want and what manufacturers can supply. Shaw's Anton was created by such a collaboration, and the U.S. Department of Energy has established three co-design centers to foster this kind of activity. At SEAS, this type of boundary-blurring collaboration happens every day in the hallways of Maxwell Dworkin and, yes, the basement of the Lyman Lab.

In 2010, the Advanced Scientific Computing Advisory Committee of the U.S. Department of Energy announced a goal—that a computing platform would be operating at exascale by 2018. The schedule has since slipped a little. Committee member Horst Simon, of Lawrence Berkeley National Laboratory, recently suggested that even 2020 may be too optimistic a target.

Murray argues that exascale machines will surely be built, despite all difficulties, "simply because the need for them is so urgent." One of the most pressing applications, she says, is "stockpile stewardship," the program of experiments and computer simulations meant to ensure the continued viability of nuclear weapons, which have now been maintained for more than 20 years without real-world testing. This program alone, Murray says, is demanding enough to ensure that at least a few exascale machines are built in the United States. She adds that certain other applications, such as climate modeling, might also be regarded as matters that impinge on national or global security.

When the first exascale computer arrives, it might not live on the Harvard campus as the Mark I did. All the same, SEAS has a major stake in these developments as contributors to the technology. Many of our research programs depend on continued progress in high-performance computing.

Perhaps most importantly, students at SEAS will help define the future of the field, inheriting a world where exascale is no longer exotic.

Provided by Harvard University

Citation: Designing exascale computers (2014, July 23) retrieved 26 June 2024 from https://phys.org/news/2014-07-exascale.html

This document is subject to copyright. Apart from any fair dealing for the purpose of private study or research, no part may be reproduced without the written permission. The content is provided for information purposes only.