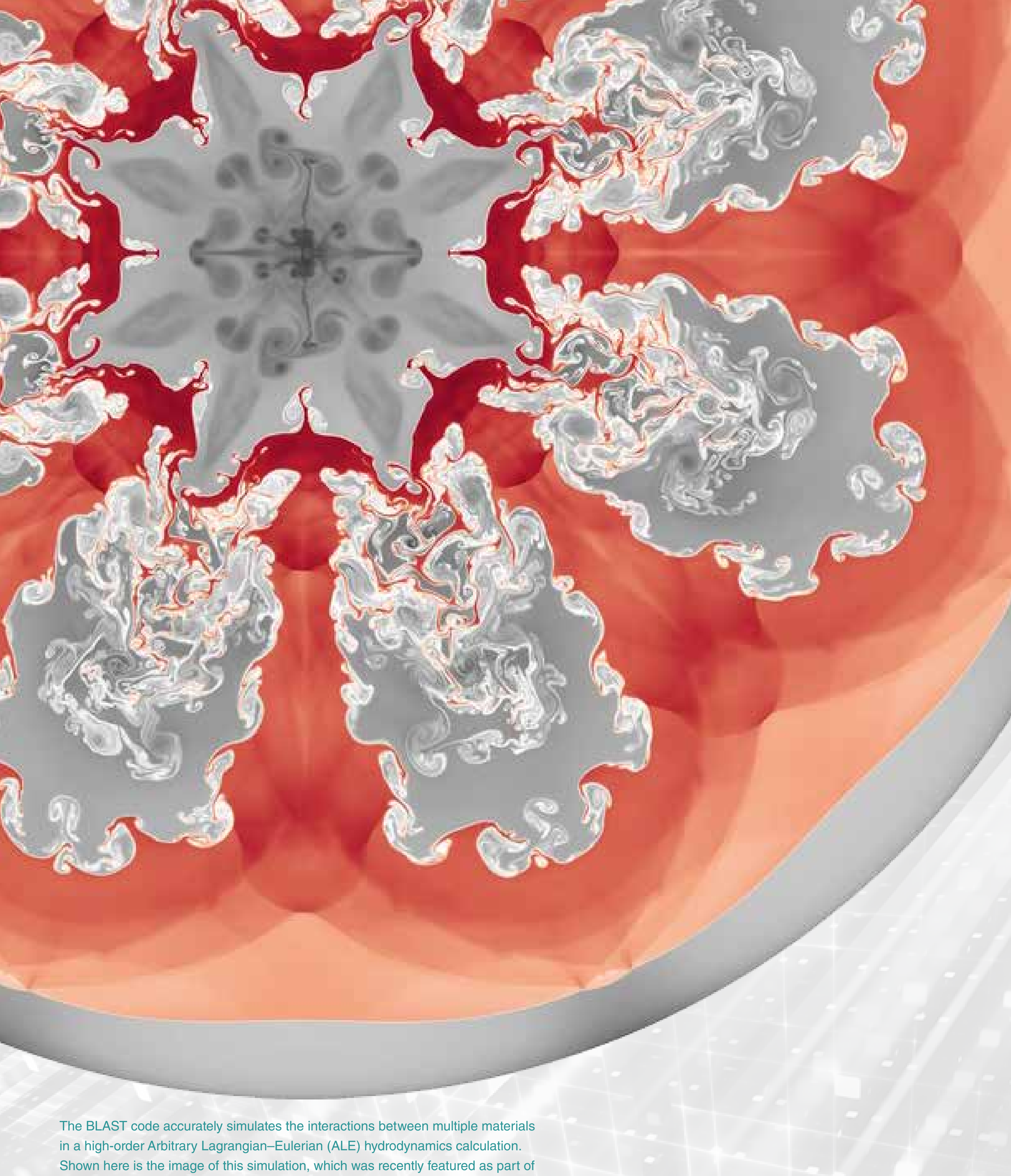


The BLAST code accurately simulates the interactions between multiple materials in a high-order Arbitrary Lagrangian–Eulerian (ALE) hydrodynamics calculation. Shown here is the image of this simulation, which was recently featured as part of an "Art of Science" exhibit at the City of Livermore's public library.

Laying the Groundwork for EXTREME-SCALE COMPUTING

Hardware, software, and code innovations are helping overcome the power and reliability challenges of next-generation supercomputers.

TO tackle many outstanding scientific and engineering challenges such as climate, energy, biosecurity, and stockpile stewardship, users need faster, massively parallel computer systems that can perform billions or trillions of calculations concurrently, access and process vast amounts of data efficiently, and run ensembles of simulations for assessing uncertainties in results. Making the transition from today's petaflop (10^{15} floating-point operations per second) systems to future exaflop (10^{18} floating-point operations per second) systems will require new computer hardware, software, and scientific codes. (See *S&TR*, March 2015, pp. 11–15.) These changes are needed to efficiently use machines with increasingly complex computer architectures, minimize power consumption, and ensure reliability.

As machines become larger and more elaborate, the need for power also grows. "We could build an exascale machine now, but it would take 100 megawatts to run it," notes Lori Diachin, manager of the Computation Directorate's project portfolio that is funded by the Laboratory Directed Research and Development (LDRD) Program. An extreme-scale—exascale or greater—system will likely contain far more processing elements than the largest present-day machines—an estimated 70–100 million cores. These elements will operate at fairly low speeds, which saves energy but increases the likelihood of performance issues, such as those related to hardware failures. (Essentially, more components increase the odds that a component will fail.) In addition, programmers seeking speed boosts must redesign their codes and algorithms to complete more tasks in parallel, maintain accuracy, and accommodate the growing number of processing elements.

The Department of Energy (DOE) has launched an ambitious effort to prepare for U.S. exascale computing within the next decade by helping to solve these complex and interrelated hardware and software challenges. As a DOE laboratory that is home to an esteemed high-performance computing (HPC) program, Lawrence Livermore is contributing to many areas of exascale research through its Advanced Simulation and Computing (ASC) Program, the DOE Office of Science's HPC programs and initiatives, as well as various LDRD-funded projects. Diachin observes, "HPC as a whole is facing a fundamental architectural shift, and these changes create a lot of uncertainty. We are using LDRD investments to explore options, develop solutions, and in some instances direct the future of computing." This article highlights six LDRD-funded research projects designed to lay groundwork for exascale computing and potential subsequent scientific breakthroughs.

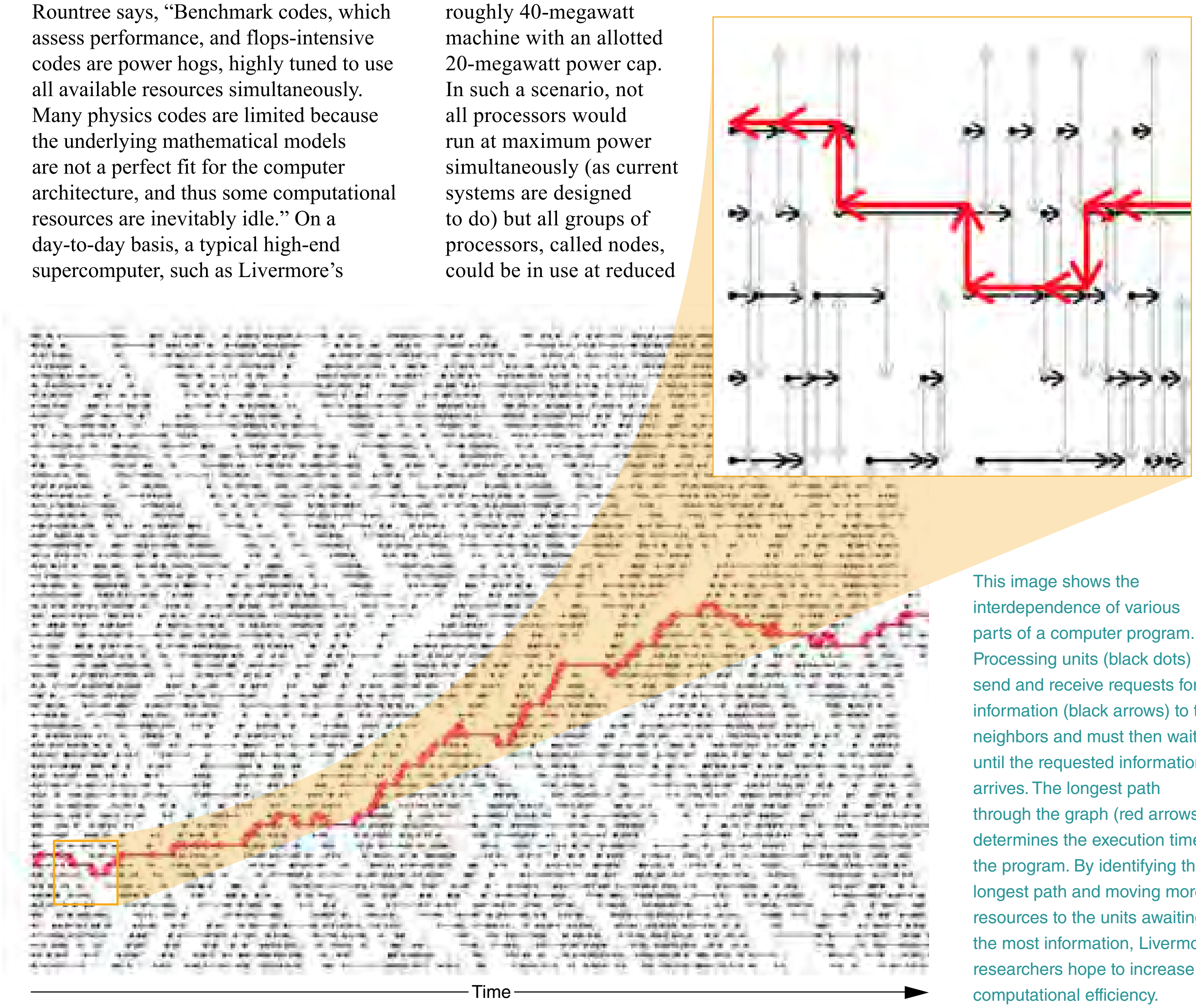
A Supercomputing Power Boost

DOE's target for exascale machine power is 20 megawatts or less—a number aimed at balancing operating costs with computing performance gains. To help achieve this goal, application developers and hardware experts are exploring ways to reduce data movement, a process that is projected to dominate the power demands of future systems. In addition, software developers are looking at new methods for monitoring and managing power.

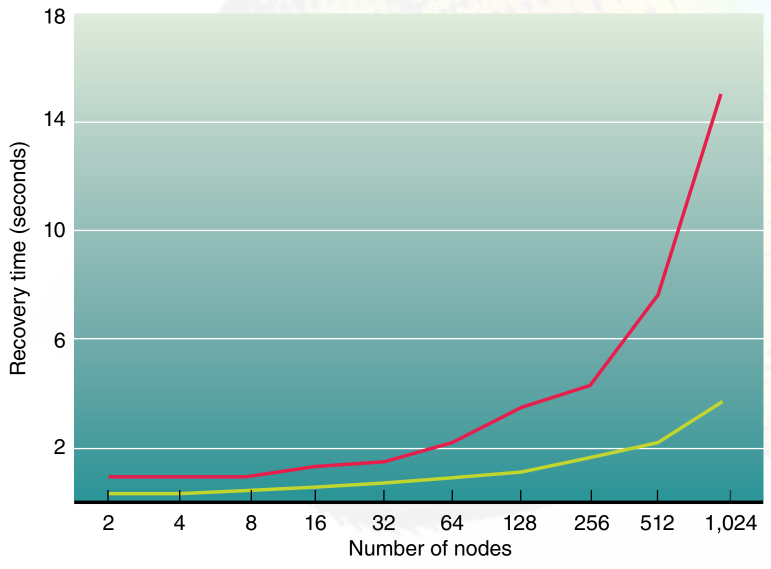
A supercomputer's flops rating represents a calculation speed (and level of capability) that the system only rarely achieves. Computer scientist Barry Rountree says, "Benchmark codes, which assess performance, and flops-intensive codes are power hogs, highly tuned to use all available resources simultaneously. Many physics codes are limited because the underlying mathematical models are not a perfect fit for the computer architecture, and thus some computational resources are inevitably idle." On a day-to-day basis, a typical high-end supercomputer, such as Livermore's Vulcan machine, uses 60 percent of its peak power.

Entirely rewriting all of Livermore's large and complex scientific codes to run more like flops-intensive ones, which complete calculations faster, is unfeasible or even impossible, according to Rountree. He and his team instead propose designing an exascale supercomputer that uses the maximum power allotment for any code it is running by overprovisioning the hardware—for example, by building a roughly 40-megawatt machine with an allotted 20-megawatt power cap. In such a scenario, not all processors would run at maximum power simultaneously (as current systems are designed to do) but all groups of processors, called nodes, could be in use at reduced power levels. Initial modeling and tests have indicated that overprovisioning can improve—even double—application performance.

Rountree's LDRD is focused on fundamental research for developing the system software to make overprovisioning practical. "The goal is not to save power—it's to use all the available power all the time," he explains. "We're looking at power as a schedulable resource. We would like to control power usage between nodes,



This image shows the interdependence of various parts of a computer program. Processing units (black dots) send and receive requests for information (black arrows) to their neighbors and must then wait idle until the requested information arrives. The longest path through the graph (red arrows) determines the execution time of the program. By identifying the longest path and moving more resources to the units awaiting the most information, Livermore researchers hope to increase computational efficiency.



Livermore computer scientists have designed and implemented Reinit, a novel backward recovery approach, to handle certain hardware failures. As indicated by recovery time measurements for failures in Livermore's Sierra machine, Reinit (green line) allows high-performance computing applications to recover up to four times faster than with traditional job restarts (red line).

within nodes, within the whole machine, and between jobs."

The team has developed and validated a model that predicts application-execution time for a given hardware configuration, accounting for factors such as the number of processors and nodes running, how tasks are divided up and completed within the processors, and how power is distributed to processors and memory. Livermore is considering this model for future one-of-a-kind, leading-edge systems.

Based on their findings thus far, the team has created two systems software tools for helping users set processor power caps and measure and monitor various power and performance indicators. These tools have been incorporated into cluster software at all three of the National Nuclear Security Administration's (NNSA's) laboratories. Next, Rountree's team will focus on how to dynamically identify and redistribute power to the slowest pieces of a calculation, which will further enhance performance.

Faster Failure Recovery

Hardware problems—referred to as faults and failures—are expected to occur more frequently in exascale systems than in petascale machines, primarily because of the new systems' greater number of processors and larger amounts of memory. (See *S&TR*, June 2012, pp. 13–15.) "Right now, our systems experience failures about once every two days. With exascale systems, failures could be as frequent as once every few hours," says computer scientist Ignacio Laguna. Ensuring that exascale systems are resilient—that hardware, software, and codes continue to function as they should despite frequent interruptions—will require a combination of hardware improvements and changes to how applications address errors and failures.

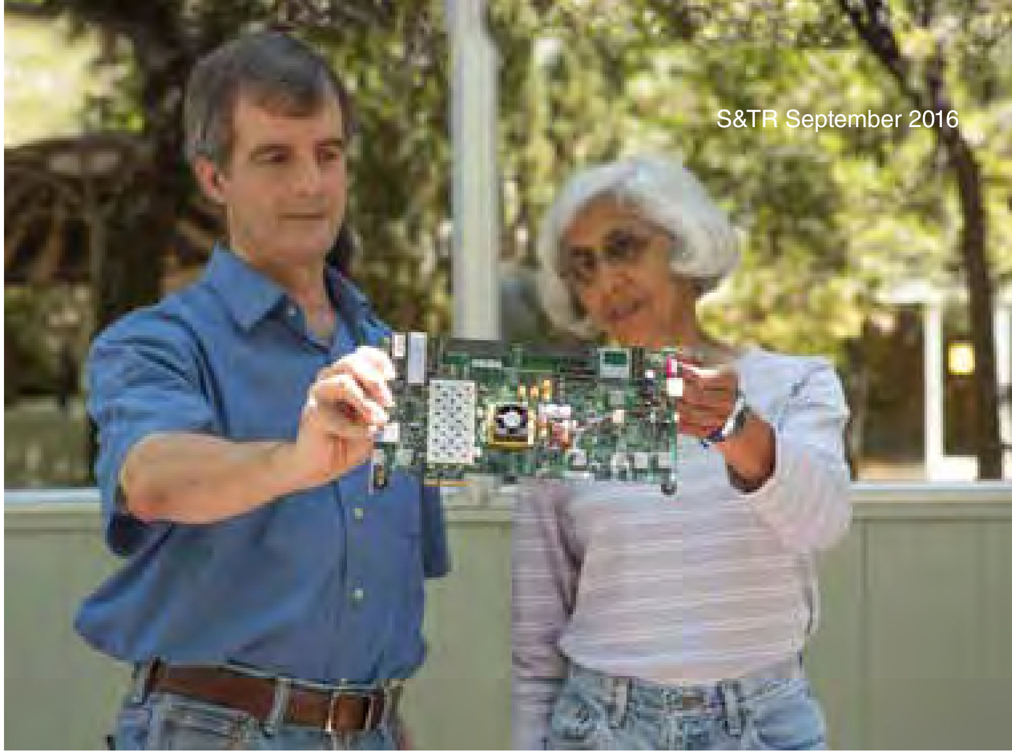
Failures are typically resolved through "backward" recovery approaches. Using this technique, a system is returned to the last checkpoint at which the data were saved and restarted from there, a process that wastes time and resources. "We want to recover from failures faster and to do so with fewer changes to the code," explains Laguna. He is leading an LDRD project to create simplified models, called programming abstractions, which allow developers to evaluate the complexity, performance, reliability, and applicability of various failure-recovery methods. The goal is to aid developers in identifying the best programming methods for meeting their recovery and performance goals.

Laguna and his team have used the Livermore-developed domain decomposition molecular dynamics code (ddcMD) to explore two types of abstractions—a backward recovery approach and a forward technique that allows the system to move ahead with a calculation using healthy, unaffected nodes. Forward recovery eliminates the need for a restart but results in a small piece of the calculation being lost, affecting accuracy. Reinit, the Livermore team's new backward approach, is an improvement when compared to existing recovery schemes. Reinit was more effective than forward recovery methods because of its easy implementation, efficiency, and broad applicability.

Laguna observes, "With HPC, speed is the measure people talk about, but productivity also matters. The time it takes to write a program is just as important as the time it takes to run it." His team will continue to explore various recovery programming models, not just forward and backward approaches. In time, they aspire to have their approaches recognized and adopted within the developer community for the message-passing interface (MPI). Used by academia, government, and industry alike, MPI is the most popular programming model for large-scale HPC.

Memory Reduces Data

Component costs and power constraints are driving a trend toward chips with more processors and correspondingly less on-chip, cache memory. Reduced local memory requires operations to fetch data from other locations to complete their calculations. The result is longer data-access times and increased stress on memory bandwidth, both of which affect code performance. Computer scientist Scott Lloyd notes, "Many



Computer scientists (from left) Scott Lloyd and Maya Gokhale inspect a memory unit emulator on a circuit board. (Photo by Lanie L. Rivera.)

newer applications have unstructured and irregular data-access patterns or other complex memory demands. These programs are much more sensitive to data-access time.” Given the far greater number of processors expected in exascale systems, without technology innovations, this memory dilemma could become a memory nightmare.

Lloyd and colleague Maya Gokhale have been exploring opportunities for reducing data movement using relatively new three-dimensional (3D) memory packaging technology, such as the Hybrid Memory Cube. These 3D units, which comprise a stack of memory layers and a basic logic layer, offer orders-of-magnitude greater bandwidth and shorter access times than is available off-chip. Additional computing functionality can be introduced on the logic layer to process data within the memory package. Gokhale and Lloyd have developed a novel method to dynamically order data to suit a code’s needs using a data rearrangement engine (DRE) in the logic layer. DRE reorganizes and filters data into local scratchpad memory—a temporary, high-speed internal memory—for use by the code. Gokhale explains, “Often, the computer fetches a whole column or row of data from memory when only part of it is needed. With a scratchpad, we can dynamically change how the data are presented for the processor.” After substantial DRE tuning and testing, the researchers demonstrated bandwidth savings, energy reductions, and up to four times faster performance, with the best performance occurring in applications featuring irregular memory-access patterns.

In-memory computing and 3D memory units are of growing interest to the HPC community. For instance, through DOE’s FastForward-2 Program, an initiative designed to accelerate the development of technologies critical to extreme-scale computing, the Livermore researchers have partnered with several vendors to explore DRE applications.

Multiple Timelines, Faster Solutions

Even though the exact architecture for exascale computers is not yet known, Livermore computational experts are preparing scientific applications, and the algorithms anchoring these codes, to run on prospective machines. As part of this effort, they are developing algorithms that account for known architectural trends such as reducing data movement and allowing for many more actions to happen in parallel, or simultaneously. The latter is particularly critical because future speedups for applications will likely happen only through greater parallelism.

Computer simulations often rely on solving time-dependent problems, such as the evolution of a supernova, the propagation of seismic waves, or the metabolism of pharmaceuticals in the human body. Until now, most solutions to these problems have addressed time steps sequentially, but this traditional time-stepping process scales up poorly on very large machines. Computational mathematicians Rob Falgout, Jacob Schroder, and their colleagues have developed a method for solving all of the time steps simultaneously with the help of a new code called XBraid—an approach that has dramatically decreased solution time for various simulations by as much as 49 times. XBraid eliminates the need for a complete code rewrite, which could be an enormous time investment. Schroder observes, “Unlike other parallel-in-time approaches, XBraid is largely nonintrusive and has a conceptually simple interface.”

The parallel-in-time approach offers a fundamentally new way to run simulations. The XBraid multigrid solver first “guesses” a solution to the problem and then uses an algorithm to generate an improved guess. This procedure repeats until the iterations converge to the final solution. The iterative process is accelerated by feeding solutions from coarser (and less computationally expensive) versions of the same problem into finer scale versions. Since the computational cost is proportional to the number of equations, a large problem can be solved in the same time as a smaller one simply by increasing the number of processors working on the calculation. Importantly, solutions from XBraid and sequential time stepping are identical, up to a user-defined tolerance.

In addition to helping researchers perform calculations more quickly, XBraid also better exploits the processing capabilities of HPC systems. "Exascale computers will be even more massively parallel than Sequoia—Livermore's 20 petaflop, 1.5-million processor flagship supercomputer," says Schroder. "We need our algorithms to make the best use of that level of parallelism."

The XBraid team is now refining and optimizing their algorithm for a range of problem types. Recent applications have included power-grid simulations and various fluid-dynamics calculations. In the future, the team will make XBraid compatible with more of the Laboratory's physics codes by exploring enhancements, such as on-the-fly refinement of spatial and temporal resolution, a significant technical challenge.

Framework Offers Greater Efficiency

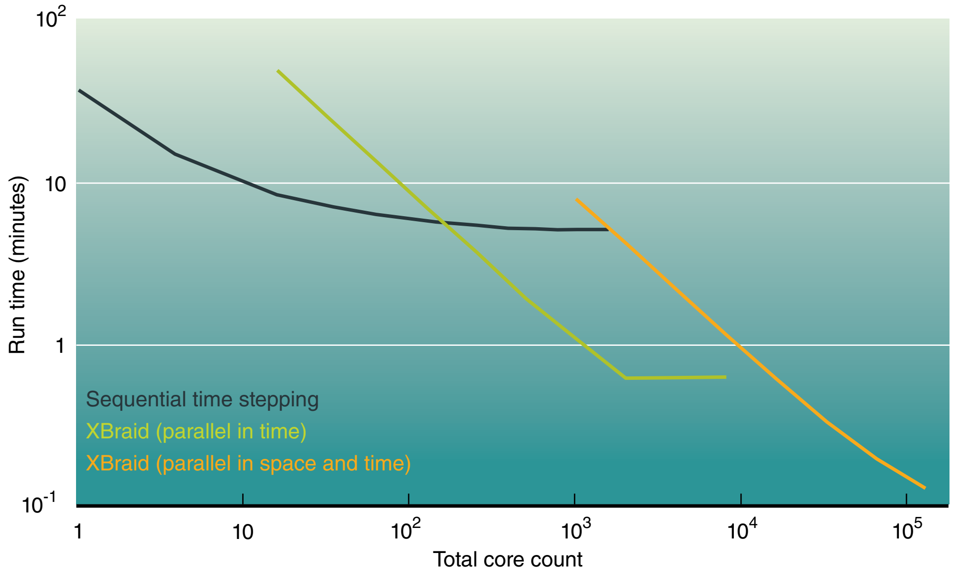
At the National Ignition Facility, researchers who study the complex shock wave interactions occurring within high-pressure experiments have found that standard low-order finite difference simulation techniques sometimes fail to satisfactorily represent material behavior under experimental conditions, particularly at material interfaces. A series of LDRD projects conducted by a team of computational mathematicians, physicists, and computer scientists led by applied mathematician Tzanio Kolev aims to remedy this issue and achieve higher fidelity simulations of shock hydrodynamics by developing more advanced numerical algorithms.

Kolev and colleagues have created a new, high-order arbitrary Lagrangian–Eulerian (ALE) framework. Standard low-order ALE methods are based on approximating the solution to a complex equation by breaking down the problem into many smaller parts (elements) along an unstructured grid, or mesh. The team's high-order ALE framework uses elements with more control points within the mesh, allowing them to curve element boundaries and the geometry inside them. "This approach helps us more accurately follow the material flow," explains Kolev. In addition, high-order methods spend less time moving data and more time crunching numbers than their lower-order counterparts, which helps offset the growing cost of data movement and enables faster calculations.

Over the past few years, the team has successfully demonstrated that the high-order ALE framework can produce accurate and robust simulations of various shock hydrodynamics problems through the method's implementation in the Livermore-developed BLAST code. The team recently incorporated a new remapping algorithm into the framework, allowing BLAST to simulate larger time steps and to more accurately model multiple materials within one element.

During a simulation, the shape of the elements in the mesh follows the movement of the simulated material, but sometimes the elements insufficiently conform to a physics field and the mesh distorts. For those portions of the calculation, the remapping algorithm "stops time" for the affected function while the mesh evolves. Once the field has been translated, along with the rest of the problem, to a more compatible mesh, the calculation continues from the point where it left off. When the mesh changes, multiple materials can be contained within the same element—a mathematically challenging situation that the BLAST team has successfully resolved, demonstrating a mix of materials can be captured at a very detailed level.

Overall, the remapping algorithm has demonstrated excellent parallel scalability, geometric flexibility, and accuracy on several model problems and single- and multiple-material ALE simulations. One of the most demanding calculations to date was a 3D BLAST simulation involving



Most solutions to time-dependent problems have addressed time steps sequentially, one at a time. The XBraid code, a parallel-in-time approach, solves all the time steps simultaneously. In this heat equation example, XBraid allows the calculation to progress up to 49 times faster.

three materials performed on 16,000 cores of Vulcan.

The team is now applying high-order solution methods to other types of physics components required for realistic multiphysics simulations, beginning with radiation diffusion. Interest in high-order methods is growing, thanks in part to the LDRD initiatives, spurring Kolev and his colleagues to propose creating a DOE high-order co-design center at Lawrence Livermore to foster greater collaboration between computer hardware, code, and algorithm developers. "To enable the high-order applications of tomorrow," he notes, "we have to ensure that we have the hardware to make these applications run well."

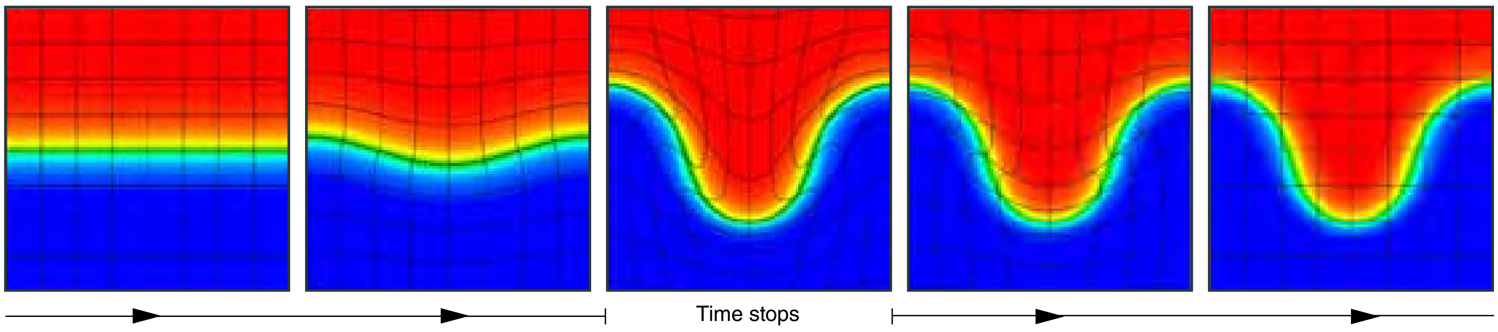
First-Principles Scaled Up

Biologists, chemists, and physicists use first-principles molecular dynamics (FPMD) codes to calculate properties of materials and molecules. These codes are based on quantum mechanics modeling, which is computationally demanding because it calculates many interactions for every atom in the problem and requires intensive communication between processors and memory. With traditional algorithms, simulation size is limited to a few hundred atoms, which is much too small for simulating complex systems or realistic materials. "The problem with FPMD methods," explains computational mathematician Jean-Luc Fattebert, "is that the cost of computing goes up faster than the size of the physical problem. Many elements of these computations are proportional to the square or cube of the system size. Given the same run time, a ten-times bigger computer will only let us solve a problem possibly twice as big."

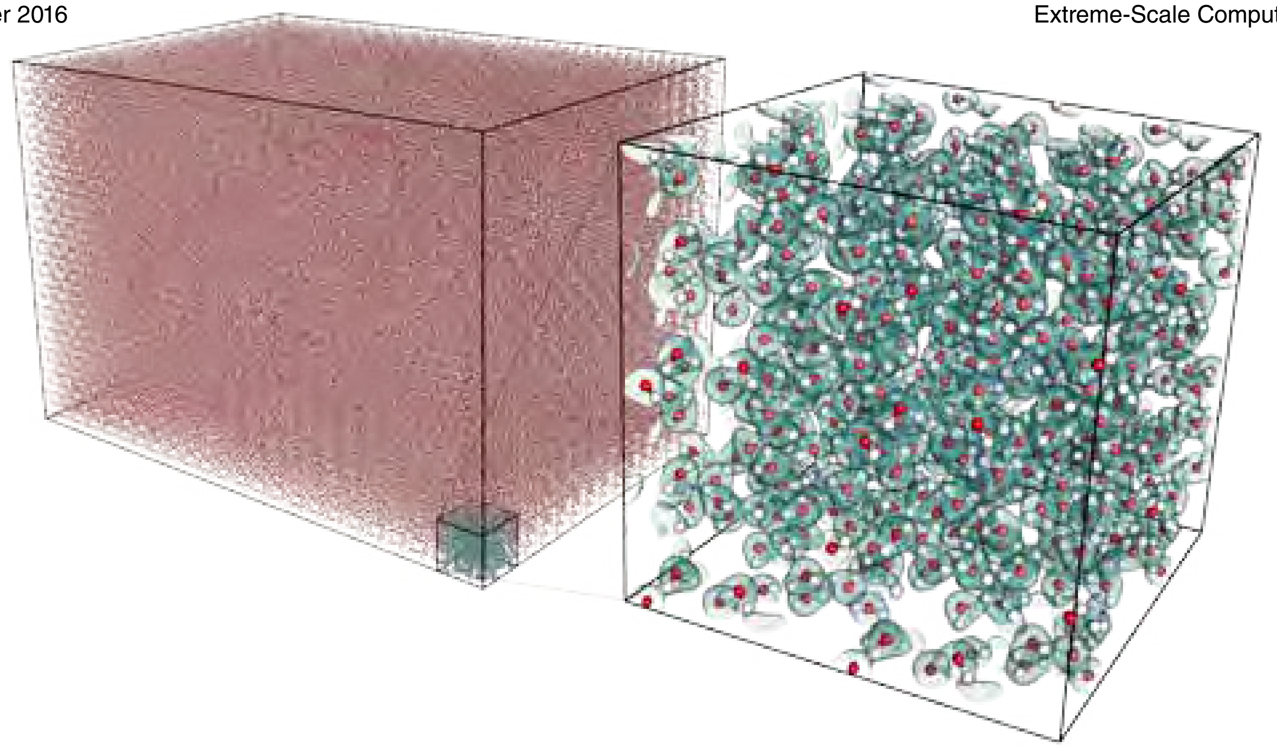
Livermore researchers, led by Fattebert, are developing new algorithms with reduced complexity and better parallel scaling to efficiently use the largest supercomputers available now and in the future to run FPMD simulations with many thousands or even millions of atoms. With these algorithms, the number of atoms that can be simulated is directly proportional to the number of processors required. The method represents only the interactions between an atom's electrons and a certain (and adjustable) number of its neighbors. In a large system, interaction across wide distances is less significant than between nearby electrons and can be safely neglected. With this approach, the number of atoms and the complexity of the systems that can be simulated go up dramatically.

The team has demonstrated excellent scaling using the novel algorithm and an accompanying new FPMD code called MGmol. For example, simulating the motion of 1,200,000 atoms for 98 time steps on roughly 1,500,000 processors of the Sequoia machine took less than 3 hours. If the team were to solve the same problem using traditional algorithms and the same number of processors, the problem would take about two years to solve and require a computer with far more memory than any currently available. This advancement earned Fattebert and his team a 2016 Gordon Bell Prize finalist spot for outstanding achievement in high-performance computing.

However, notes Fattebert, "It is one thing to show that we scale, but we also have to show that we are not sacrificing accuracy," as has occurred in many previous attempts to speed up FPMD calculations. Fortunately, validations of their solutions for nonmetallic systems, used for studying problems such as the dilute solutions in energy-storage devices, are progressing well. For instance, when calculating a problem involving 512 water molecules, both MGmol and Livermore FPMD code QBox, which uses more traditional algorithms, produced virtually identical results. The researchers have also recently begun developing similar methods for simulating metallic systems—a more challenging problem because fewer electron interactions can be ignored,



When the shape of grid elements in a simulation insufficiently conforms to a physics field, the BLAST remapping algorithm "stops time" (center) and institutes a remap phase, wherein the field stays the same while the mesh evolves. The field, along with the rest of the problem, is then translated to a new mesh, and the simulation is restarted from where it left off.



making computational cost reductions more difficult to achieve.

In the realm of metals, very large and accurate simulations will help researchers create better alloys. At low concentrations in particular, the percentage contribution of a given component can dramatically change an alloy's physical properties. Fattebert is also looking down the road to the next algorithmic and computational challenge: lengthening the duration of simulations to enable studies of slower processes, or a longer stretch of a specific process, thereby making FPMD a practical solution for more researchers, including biologists.

Beyond Exacale

Exascale is the crucial next step in the evolution of HPC, but it is far from the final one. Livermore researchers have already begun planning for more radical shifts in computing. "The Laboratory is making 'beyond Moore's Law' investments," says Diachin. "At some point, we cannot get more speed by making transistors smaller and putting more on a chip, as there's a physical limit to the technology. The question is, what will architectures look like when these limits are reached?"

One such investment by the ASC Program supports research on neuromorphic, or brain-inspired, computing systems that can perform tasks such as pattern recognition and image processing more efficiently than standard computers. (See *S&TR*, June 2016, pp. 16–19.) The Laboratory acquired its first neuromorphic platform from IBM in March 2016. (See *S&TR,* September 2016, p. 2.) Also, several LDRD projects have been exploring how to make quantum computing practical. This type of computing exploits the laws of quantum mechanics to complete tasks, in some instances far more quickly than standard computers. Bringing these and other exotic computing concepts to fruition will take patience, creativity, collaboration, and dedication—characteristics Livermore researchers have already abundantly demonstrated in their efforts to build a foundation for exascale computing.

—*Rose Hansen*

Key Words: Advanced Simulation and Computing (ASC) Program, algorithm, arbitrary Lagrangian–Eulerian (ALE), BLAST code, data rearrangement engine (DRE), domain decomposition molecular dynamics (ddcMD) code, exascale, extreme-scale computing, FastForward-2 Program, first-principles molecular dynamics (FPMD), floating-point operations per second (flops), Laboratory Directed Research and Development (LDRD) Program, Message-Passing Interface (MPI), MGmol code, neuromorphic computing, programming abstraction, QBox code, quantum computing, Reinit code, Sequoia, Sierra, three-dimensional (3D) memory, Vulcan, XBraid code.

Livermore's MGmol code and a novel algorithm made possible a massive first-principles molecular dynamics simulation involving 1,179,648 atoms (393,216 water molecules). The simulation was performed on Livermore's Sequoia machine using 1,572,864 processors. The inset, showing a small subset of the simulation, illustrates the challenge of representing all the atoms accurately. This simulation would not be feasible using traditional algorithms. (Image courtesy of Liam Krauss.)

For further information contact Lori Diachin (925) 422-7130 (diachin2@llnl.gov).