The Impact of Heterogeneous Computer Architectures on Computational Physics

Barry I. Schneider | National Institute of Standards and Technology

High-performance computing has seen many changes in the past decade. One such change, conditioned by both the slowing down of Moore’s law and also by the need to reduce the power requirements of compute processors, has been to develop especially designed chips/cards to accelerate many of the operations that are common to the types of computations needed in physics, chemistry, and engineering calculations. Some of these computational accelerators, such as the NVIDIA graphical processing units, or GPUs, were initially developed for a completely different purpose. It was cleverly found that the GPUs could also be adapted to scientific computation, although it took a good deal of effort to realize the inherent potential in that technology. In somewhat more recent times, other technologies, such as the Intel Phi, have been incorporated into modern supercomputers. It’s probably fair to say that these developments will continue as scientists seek to develop more sophisticated computational models that will place accelerating demands on the ability of computer chips to perform floating-point computations.

While these developments are welcome, they come with some nontrivial costs. These accelerators
often require new programming paradigms even when more traditional computing languages can be used, and straightforward application of the old, tried, and true approaches can fail. There's little doubt that the community is facing serious challenges to obtain really exceptional performance from these devices. Where serious attempts have been made by computational scientists to find the most efficient approaches to utilizing these “beasts,” we often see remarkable improvements in performance. In addition, it's often the case that the re-programming for the accelerators improves the performance of codes on standard CPUs. Because it's more than likely that the supercomputers of the future will be built using many different chip technologies, heterogeneous computing is here to stay. Thus, the scientific community will have to adapt to using these technologies if it is to remain at the bleeding computational edge and tackle the complex problems required to push the boundaries of science and engineering.

About a year ago, I felt that the time was ripe to organize a session at the American Physical Society's March and April meetings to expose the more general physics community to what has been happening with these accelerators in several subfields of physics and chemistry. I felt it would be worthwhile to hear from the folks in the trenches about what was needed to achieve good performance from these devices. My hope was that those who attended would not only see what can or can't be done by using these new technologies but more importantly get a sense of what common practices need to be employed so that this doesn’t result in having to start from ground zero with each new development likely to appear on the horizon. If the computational science community has to continually reprogram its codes every time a new device appears, it’s clearly a losing battle in terms of available manpower.

The sessions were successful and reasonably well attended, but I wanted a more permanent record of what was presented. As an editor for CiSE, I felt this seemed like the perfect place for a series of articles on what has been done and what the future holds for heterogeneous computing. I spoke to all the participants in the symposia about writing an article that would give a more expansive view of what they presented at the meeting. The current issue contains the result of that endeavor. The next few sections contain a summary of the articles and some comments on the material presented.

**Lattice Quantum Chromodynamics (QCD)**

High-energy physicists were among the earliest users of supercomputers and still remain at the bleeding edge in their use of emerging new technologies.

Andrei Alexandru, in “Lattice Quantum Chromodynamics with Overlap Fermions on GPUs,” gives a nice overview of how GPUs have influenced QCD in applications to nuclear physics. Using multiple GPUs within what's known as the overlap operator model, which importantly preserves chiral symmetry, we need to propagate the Hamiltonian in imaginary time, discretizing the problem on a lattice of about \(2 \times 10^6\) sites. The article focuses in particular on the tools for the analysis of the overlap fermion model. In this model, we need to invert a matrix that's sensitive to eigenvalues near zero. Alexandru demonstrates that explicit removal of these enables the very large number of inversions to be done quite efficiently. The use of multiple GPUs is demonstrated, and the article points to all the positive and negative issues associated with using GPUs for this application.

Colin Morningstar, in “Unearthing Excited Hadron Resonances in Lattice QCD Using NSF XSEDE Resources,” concentrates on the evaluation of the Dirac matrix (\(K\)) using sophisticated Markov-chain Monte Carlo methods. Given the huge size of the matrix, it isn't possible to exactly compute or even store the matrix elements. By using the so-called “point-to-all” method, the number of matrix elements required in the computation can be drastically reduced. Although Morningstar is only beginning to adapt his approach to GPU and Xeon Phi accelerators, the article contains a lot of information on the numerical methods needed to compute the quark propagators. Many of these methods are common to all approaches, reducing the physical problem to the solution of a large set of algebraic equations; Krylov and restarted Lanczos approaches reign. The author points to the value of the NSF XSEDE project and the computational facilities at the Texas Advanced Computing

The scientific community will have to adapt to using these technologies if it is to remain at the bleeding computational edge.
Center (Stampede) and the National Institute for Computational Science (NICS-Kraken) as crucial in enabling the science in his group.

**Molecular Dynamics**
In 1959, Berni Alder and Thomas Wainwright published what is considered the first paper on molecular dynamics (MD). It’s understandable that the work was done at Lawrence Livermore National Laboratory because it was one of the few places with high-speed computational resources at the time. In late 1964, Aneesur Raman independently discovered the power of MD to understand the correlations of atoms in liquid Ar. The work examined 864 Ar atoms interacting via Lennard-Jones potentials. The calculation was performed on a CDC 3600 computer at Argonne National Laboratory. Since those early days, MD has seen application to a broad array of problems in physics, chemistry, and engineering.

An article by Thomas Cheatham and his colleagues, “The Impact of Heterogeneous Computing on Workflows for Biomolecular Simulation and Analysis,” describes recent work on the impact of heterogeneous computing on workflows for biomolecular simulation and analysis. Today, detailed and accurate biomolecular simulation is not only possible but has become routine. As Cheatham notes, the rate limiting steps of common biomolecular simulation workflows has shifted to the pre- and postprocessing phases of the workflow, namely, simulation set-up and data processing, management, and analysis. The article describes in some detail the history of the use of accelerators in MD, and the impact of GPUs on the current generation of MD codes, in particular AMBER and the ANTON computer developed by D.E. Shaw Research specifically designed for MD simulations. The article discusses many questions that have arisen in recent years such as whether equivalent results can be obtained comparing one long simulation on Anton to a series of independent and aggregated simulations performed on GPU resources. The article concludes that current and future heterogeneous computational resources are required in order to facilitate the efficient analysis of increasingly large MD simulations.

**Cosmology**
Simulations of the entire universe are arguably models of the largest and most complex physical system that exists. The first cosmological simulations were carried out over 40 years ago with a mere 300 particles. Today, the models have grown to include up to a trillion particles plus far more sophisticated physics. Rupert Croft and his colleagues, in “Petascale Cosmology: Simulations of Structure Formation,” provide a comparatively short description of their recent work with full-machine runs on Blue Waters, the largest of the National Science Foundation’s supercomputers, using the P-Gadget code. These smooth particle hydrodynamic (SPH) simulations are among the largest ever run, and they have reached their initial target redshift, the end of the reionization epoch. To capture the large scale gravitational forces that drive the growth of structure, the simulation volumes must be at least tens to hundreds of megaparsecs in size. As a result, even with hundreds of billions of particles, the particle masses are necessarily much larger than individual stars. The particle mass in these simulations was around a million solar masses. Subgrid models are employed for star formation and for the formation and growth of supermassive black holes. The article spends a significant amount of time discussing the numerical methods and computational challenges.

**Plasma Simulations**
In “Skeleton Particle-in-Cell Codes on Emerging Computer Architectures,” Viktor Decyk discusses how emerging computer architectures are impacting particle-in-cell plasma codes. He casts things in a much more general framework, beginning with the assumption that the next generation of supercomputers will consist of a hierarchy of parallel computers. If each supercomputer node is then viewed as a parameterized abstract machine, it’s possible to design algorithms independently of the hardware. Such an abstract machine can be defined to consist of a collection of vector (SIMD) processors, each with a small shared memory, communicating via a larger global memory. Such an abstraction fits a variety of hardware, including GPUs, Phis, and multicore processors with vector extensions, and is the basis of the OpenCL programming language. Both old and new ideas are necessary to implement this concept efficiently. The author illustrates this approach using Plasma PIC codes. The PIC codes were developed to be

---

**Constantly evolving architectures are placing severe demands on both computer and computational scientists.**
adaptable to architectures other than the GPUs but evolved from experimentation done at UCLA to efficiently port PIC codes to NVIDIA GPUs. An important principle emerged: optimizing data movement is crucial to performance. You need to avoid accesses to global memory as much as possible by copying data to fast local memory. The resulting challenge was overcome by sorting particles into partitions small enough so that they could fit into local fast memory. Thus the particles needed to be reordered at every time step, a challenge discussed in the article. The author concludes, quite optimistically, that the scientific community will emerge from this revolution as successfully as it has from past ones.

**High-Performance Scientific Software**

Developing application codes that perform efficiently on heterogeneous, multinode, multicore systems with deep memory hierarchies is a daunting task. Erik Schnetter and his colleagues, in "Chemora: A PDE Solving Framework for Modern HPC Architectures," describe a framework for solving partial differential equations (PDEs) that have been discretized using finite difference, finite element, adaptive mesh, or multiblock approaches. Chemora is based on Cactus and has been implemented in the Einstein Toolkit. PDEs are expressed in high-level languages such as LaTeX or Mathematica and the computational stencils defined separately from the equations. The goals are to assist the design of large applications software recognizing that hardware and languages change and that MPI and OpenMP are difficult to use at large core counts. The Cactus modules/libraries or thorns may be combined together by the developer in his or her specific application, which gives flexibility but also requires user input. The article goes on to describe the Einstein Toolkit and shows a simple example of three coupled PDEs treated using simple finite differencing. The code generation method—Krancke—which is based on Mathematica to describe the equations. The Kranc package expands the equations into (say) C++ code, which then becomes a completely independent Cactus thorn. The article goes into a significant amount of detail on many aspects of the project such as single-node performance, multithreading, and improving communication performance. The authors conclude with an example of core collapse supernova. Finally, performance comparisons are shown between Blue Waters and a local machine on the Einstein equations and reasons for the differences discussed.

**Computer Architecture**

The final two articles in this special issue of *GiSE* are devoted to discussions about current and emerging hardware trends. R. Glenn Brook and his colleagues, in "Beacon: Exploring the Deployment and Application of Intel Xeon Phi Coprocessors for Scientific Computing," give us a snapshot of the NSF Beacon project, which is exploring the impact of the new Intel Xeon Phi coprocessor for a variety of scientific and engineering applications. Beacon is a Cray CS300-AC with 48 compute nodes each with two Intel CPUs and four Intel Xeon Phi's. With 768 conventional and 11,520 accelerator cores, the machine has a peak of 210TF. A key feature of the machine is its energy efficiency (first in the Green500 list) compared to a comparable more standard architecture. The article goes into some detail about the configuration of the cluster and the challenges to its users in achieving good performance. Software challenges are also mentioned, and the article describes how the team responded to those challenges. Three large software applications were discussed: GROMACS, a large MD simulations package; SG++, a data mining application; and SeisSol, an earthquake simulation code. For GROMACS, the Phi performed about 50 percent better than the single GPU implementation. The other applications also had performance improvements. Of course, none of this comes without a serious investment in software re-engineering. But the enhanced computational performance and energy efficiency attained by such emerging architectures are successfully enticing computing centers and computational researchers alike to adopt them for demanding applications.

The article by Jeffrey Vetter, "Opportunities for Nonvolatile Memory Systems in Extreme-Scale High Performance Computing," examines the question of system-wide power consumption as we move into the era of extreme-scale computing. DRAM accounts for 30 to 50 percent of the power consumption of a node and the difficulty of balancing memory capacities with increasing computational rates is a substantial challenge. Several emerging memory technologies are being investigated as an alternative for DRAM, some of which might offer solutions for HPC architectures as Vetter discusses in some detail in the article. Researchers are investigating how to integrate these new technologies into future extreme-scale HPC systems and how to expose these capabilities to their applications. Current results show that
several of these strategies may provide more main memory capacity at the same or reduced power costs and offer both higher bandwidth I/O and new opportunities for application resilience and output postprocessing, such as in situ visualization and incremental checkpointing.

The world of high-performance computing is changing at a rapid pace. Constantly evolving architectures are placing severe demands on both computer and computational scientists. This special issue provides a snapshot of the current state of the art; hopefully, it also provides some insights as to where the current lessons learned can be used in the future to simplify the extremely large burden needed to make large-scale scientific applications run efficiently on future hardware.

Acknowledgments
Many people have worked tirelessly to bring this special issue to fruition. I thank the authors for their great effort in making their work accessible and understandable to a reasonably broad audience. The reviewers, who have spent a lot of time reading the manuscripts and suggesting improvements, also must be congratulated on their efforts. Finally, the CiSE staff at all levels has been flexible and extremely helpful in getting this issue to press on time.

Barry I. Schneider is a senior research scientist at the National Institute of Standards and Technology. He’s also the associate editor of the NIST Digital Library of Mathematical Functions, a Fellow of the American Physical Society (APS), a Humboldt Prize winner, and vice chair of the Division of Computational Physics for the APS. He has been involved in developing a broad array of scientific software applications in atomic and molecular physics and quantum chemistry for over 45 years. Contact him at bis@nist.gov.

Selected articles and columns from IEEE Computer Society publications are also available for free at http://ComputingNow.computer.org.