23 Global Gyrokinetic Particle-in-Cell Simulation

William Tang and Zhihong Lin

CONTENTS

23.1 INTRODUCTION

As the global energy economy makes the transition from fossil fuels toward cleaner alternatives, fusion becomes an attractive potential solution for satisfying the growing needs. Fusion energy, which is the power source for the sun, can be generated on earth, for example, in magnetically confined laboratory plasma experiments (called *tokamaks*) when the isotopes of hydrogen (e.g., deuterium and tritium) combine to produce an energetic helium *alpha* particle and a fast neutron—with an overall energy multiplication factor of 450:1. Building the scientific foundations needed to develop fusion power demands high-physics-fidelity predictive simulation capability for magnetically confined fusion energy (MFE) plasmas. To do so in a timely way requires utilizing the power of modern supercomputers to simulate the complex dynamics governing MFE systems—including International Thermonuclear Experimental Reactor (ITER), a multibillion dollar international burning plasma experiment supported by seven governments representing over half of the world's population. Currently, under construction in France, ITER will be the world's largest tokamak system, a device that uses strong magnetic fields to contain the burning plasma in a doughnut-shaped vacuum vessel. In tokamaks, unavoidable variations in the plasma's ion temperature profile drive microturbulence—fluctuating electromagnetic fields, which can grow to levels that can significantly increase the transport rate of heat, particles, and momentum across the confining magnetic field. Because the balance between these energy losses and the self-heating rates of the actual fusion reactions will ultimately determine the size and cost of an actual fusion reactor, understanding and possibly controlling the underlying physical processes is key to achieving the efficiency needed to help ensure the practicality of future fusion reactors. The associated motivation drives the pursuit of sufficiently realistic calculations of turbulent transport that can only be achieved through advanced simulations. The present paper on advanced particle-in-cell (PIC) global simulations of plasma microturbulence at the extreme scale is accordingly associated with this fusion energy science (FES) grand challenge [1,2].

The research and development (R&D) described in this document targets new physics insights on MFE confinement scaling by making effective use of powerful world class supercomputing systems. Specifically, the long-time behavior of turbulent transport in ITER scale plasmas is studied using simulations with unprecedented phase-space resolution to address the reliability/realism of the well-established picture of the influence of increasing plasma size on confinement in tokamaks and the associated physics question of how/if the aforementioned turbulent transport changes with the size of laboratory plasmas up to the ITER scale.

Associated knowledge gained addresses the key question of how turbulent transport and associated confinement characteristics scale from present generation devices to the much larger ITER plasmas. This involves the development of modern software capable of using leadership class supercomputers to carry out reliable first-principles-based simulations of multiscale tokamak plasmas.

Particle dynamics can in general be described either by a five-dimensional (5D) gyrokinetic equation (for low-frequency turbulence) or six-dimensional (6D) fully kinetic equation (for high-frequency waves). The flagship gyrokinetic toroidal code (GTC) and its *codesign* partner GTC-P are massively parallel PIC codes designed to carry out first principles, integrated simulations of thermonuclear plasmas, including the future burning plasma ITER. These codes solve the 5D gyrokinetic equation in full, global toroidal geometry to address kinetic turbulence issues in magnetically confined fusion experimental facilities. GTC is the key production code for the fusion SciDAC project Gyrokinetic Simulation of Energetic Particle Turbulence and Transport (GSEP) Center and the Accelerated Application Readiness (CAAR) program at the Oak Ridge Leadership Computing Facility (OLCF). It is the only PIC code in the world fusion program capable of multiscale simulations of a variety of important physics processes in fusion-grade plasmas including microturbulence, energetic particle dynamics, collisional (neoclassical) transport, kinetic magnetohydrodynamic (MHD) modes, and nonlinear radio-frequency (RF) waves. GTC interfaces with MHD equilibrium solvers (e.g., in codes such as equilibrium and reconstruction fitting code (EFIT), VMEC, and M3D-C1) for addressing realistic toroidal geometry features that include both axisymmetric tokamaks and nonaxisymmetric stellarators. A recent upgrade enables this code to carry out global PIC simulations covering both the tokamak core and scrape-off layer (SOL) regions. It should also be noted that the current flagship version of GTC is capable of both perturbative (δf) and nonperturbative (full-f) simulations with capability of dealing with kinetic electrons, electromagnetic fluctuations, multiple ion species, collisional (neoclassical) effects using Fokker–Planck collision operators, equilibrium current and radial electric field, plasma rotation, sources/sinks and external antennae for auxiliary wave heating.

Beyond the conventional application domain of gyrokinetic simulation for microturbulence, the GTC code has a long history in pioneering the development and application of gyrokinetic simulations of mesoscale electromagnetic Alfven eigenmodes excited by energetic particles (EP) in toroidal geometry. This is one of the most important scientific challenges that must be addressed in future burning plasma experiments such as ITER. Accordingly, the GTC work-scope has recently been extended to include simulation of macroscopic kinetic MHD modes driven by equilibrium currents. The associated importance is that such efforts could ultimately lead to key knowledge needed to systematic analyze and possibly help avoid or mitigate highly dangerous reactor relevant thermonuclear disruptions.

23.2 SCIENTIFIC METHODOLOGY

The GTC and GTC-P codes include all of the important physics and geometric features captured in numerous global PIC simulation studies of plasma size scaling over the years—extending from the seminal work in the *Phys. Rev. Letter* (PRL) by Lin et al. [3] up to the more recent PRL paper by McMillan et al. on "system size effects on gyrokinetic turbulence" [4]. The current generally supported picture is that size-scaling follows an evolution from a *Bohm-like* trend where the confinement degrades with increasing system size to a *Gyro-Bohm-like* trend where the confinement for Joint European Torus (JET)-sized plasmas begins to *plateau* and then exhibits no further confinement degradation as the system size further increases toward ITER-sized plasmas. A number of physics papers over the past decade have proposed theories—such as turbulence spreading—to account for this transition to gyro-Bohm scaling with plasma size for large systems. From a physics perspective, the main point in this paper is that this key decade-long fusion physics picture of the transition or *rollover* trend associated with toroidal ion temperature gradient microinstabilities that are highly prevalent in tokamak systems—should be re-examined by modern supercomputing-enabled simulation studies, which are now capable of being carried out with much higher phase-space resolution and duration. With a focused approach based on performance optimization of key functions within PIC codes in general, GTC-P, the *codesign* focus, has demonstrated the effective usage of the full power of current leadership class computational platforms worldwide at the petascale and beyond to produce efficient nonlinear PIC simulations that have advanced progress in understanding the complex nature of plasma turbulence and confinement in fusion systems for the largest problem sizes. Unlike fluid-like computational fluid dynamics (CFD) codes, GTC-P has concentrated on the fact that PIC codes are characterized by having less than 10 key operations, which can then be an especially tractable target for advanced computer science performance optimization methods. As illustrated in Ref. [5], these efforts have resulted in accelerated progress in a discovery-science-capable global PIC code that models complex physical systems with unprecedented resolution and produces valuable new insights into reduction in *time-to-solution* as well as *energy-to-solution* on a large variety of leading supercomputing systems.

The *flagship* GTC code is—as noted earlier—especially comprehensive with respect to the complex physics included. Its productivity over the years is well illustrated in Table 23.1.

TABLE 23.1
Demonstration of GTC Productivity and Impact → Delivery of Scientific Advances with Use of Increasingly Powerful Supercomputing Systems

GTC Simulation	Computer Name	PE # Used	Speed (TF)	# Particles Used	Time Steps	Physics Discovery (Publication)
1988	Cray T3E NERSC	10^2	10^{-1}	10^8	10^4	Ion turbulence zonal flow (***Science**, 1998*)
2002	IBM SP NERSC	10^3	10^0	10^9	10^4	Ion transport size scaling (***PRL**, 2002*)
2007	Cray XT3/4 ORNL	10^4	10^2	10^{10}	10^5	Electron turbulence (***PRL**, 2007*); EP transport (***PRL**, 2008*)
2009	Jaguar/Cray XT5 ORNL	10^5	10^3	10^{10}	10^5	Electron transport scaling (***PRL**, 2009*); EP-driven MHD modes
2012 to present	Cray XT5 Titan ORNL Tianhe-1A (China)	10^5	10^4	10^{11}	10^5	Kinetic-MHD (***PRL**, 2012*); Turbulence + EP + MHD TAE Modes (***PRL**, 2013*)
2018 (future)	Path to Exascale HPC Resources	TBD	10^6	10^{12}	10^6	Turbulence + EP + MHD + RF

*** GTC is first FES code to deliver production run simulations @ TF in 2002 and PF in 2009.

Several key associated computational methodologies will be elaborated upon as follows.

23.2.1 Kinetic Electron Models

The small electron mass presents a numerical difficulty for simultaneously treating the dynamics of ions and electrons in long time simulations. A fluid-kinetic hybrid electron model [6] currently implemented in GTC overcomes this difficulty by expanding the electron drift kinetic equation using the electron–ion mass ratio as a small parameter. The model accurately recovers low-frequency plasma dielectric responses and faithfully preserves linear and nonlinear wave-particle resonances. Maximum numerical efficiency is achieved by overcoming the electron Courant condition and suppressing tearing modes and high-frequency modes thus effectively suppressing electron noise. The fluid-kinetic hybrid electron model avoids the well-known *cancellation problem* in some gyrokinetic particle and continuum codes [7]. The *cancellation problem* arises when solving a particular form of the Ampere's law, where two large terms are artificially added to the original Ampere's law. These two terms are needed because canonical momentum is used as an independent variable to overcome a numerical difficulty of calculating the inductive electric field by an explicit time derivative. Analytically, these two terms should cancel with each other exactly. However, a small error in numerically evaluating these two large terms can give rise to a residue, which leads to a large error in solving the Ampere's law.

Moving into the near future, the very high-resolution capability in advanced PIC codes (such as GTC-P) hold strong promise of being further improved on the 200-petaflop near future leadership-class systems such as Summit and Aurora. Accordingly, progress toward delivering the skin-depth grid resolution capability [8] to comprehensively avoid the aforementioned *cancellation problem* in fully kinetic electromagnetic codes will likely be realistically achievable with access to the 200 PF class of supercomputers.

23.2.2 Resistive Tearing Mode Model

The fluid-kinetic hybrid electron model incorporating equilibrium current enables global gyrokinetic PIC simulations of both pressure-gradient-driven and current-driven instabilities—as well as their nonlinear interactions in multiscale simulations [9]. However, the fluid-kinetic hybrid electron model removes the tearing modes in order to improve numerical properties for facilitating production runs. In order to simulate classical tearing modes, the current GTC capabilities have recently been extended for this model with the inclusion of a resistivity term (due to friction between electrons and ions) in the electron momentum equation

$$\frac{\partial \delta A_{||}}{\partial t} = -c\mathbf{b}_0 \cdot \nabla \delta\phi + \frac{cT_e}{e}\mathbf{b}_0 \cdot \nabla\, \partial n_e + \eta \frac{c^2}{4\pi} \nabla_\perp^2 \delta A_{||}$$

This nonideal term introduces the resistive tearing mode physics into the GTC formulation [10].

23.2.3 Collisionless Tearing Mode Model

Collisionless tearing mode (e.g., microtearing mode), which is not considered important for conventional tokamak plasmas such as ITER, has been observed in some high-spherical tokamaks such as National Spherical Torus Experiment (NSTX), Mega Amp Spherical Tokamak (MAST). To simulate the collisionless tearing mode, we have implemented in GTC two new kinetic electron models: the split-weight scheme that treats the full physics of the electron drift kinetic equation including the collisionless tearing mode, and the finitemass electron fluid model. In the finite-mass electron fluid model, the electron momentum equation in the original fluid-kinetic hybrid electron model is upgraded to include the electron inertia term using the following electron parallel momentum equation:

$$\frac{\partial\delta\xi}{\partial t}=\frac{\omega_{pe}^2}{c}\left(\frac{T_e}{n_0 e}\mathbf{b}_0\cdot\nabla\delta n_e-\mathbf{b}_0\cdot\nabla\delta\phi\right)+\frac{4\pi e}{m_e c}\frac{\boldsymbol{\delta B}_\perp}{B}\cdot\nabla n_0 T_e$$

Here, $\delta\xi=-\left(\nabla_\perp^2=-\frac{1}{D_e^2}\right)\delta A_{||}$. The rest of the set of equations of the original fluid-kinetic hybrid electron model remain unchanged. Electron kinetic effects appear in higher order equations by using the nonadiabatic part of the drift-kinetic equation. This finite-mass kinetic-fluid hybrid electron model has been verified for simulating the theoretically predicted collisionless tearing mode instability in the slab geometry [11] and has been recently implemented and verified in GTC [12].

The formulation of the resistive and collisionless tearing has been combined to simulate the transition from collisionless to resistive tearing mode when the resistivity is increased, and recover the correct scaling of the collisionless tearing mode growth rate on the skin depth when the skin depth is much shorter than the macroscopic length. This complete formulation will be used for self-consistent simulation of the interactions between microturbulence, tearing mode, and neoclassical transport proposed in this project. This fluid electron models for collisionless tearing mode has recently been extended and verified to fully incorporate electron kinetic effects using the fluid-kinetic electron models, which is being applied to study the microtearing mode in high beta plasmas.

23.2.4 Global PIC Geometric Models

In plasma turbulence studies, the standard approach is to divide the physical quantities into an equilibrium part and a fluctuating part. The GTC code uses two set of meshes—one for the specification of the equilibrium and the other to represent fluctuating turbulent fields. In particular, the turbulence mesh is an unstructured field-aligned mesh for finite difference or finite element in three-dimensional (3D) space.

The equilibrium quantities are governed by the Grad–Shafranov equation for toroidal geometry, while the fluctuating part is driven by various instabilities that lead to turbulent transport. Equilibrium magnetic configurations typically used in gyrokinetic simulations come from (1) analytic models such as the simple circular cross section or the Miller equilibrium and (2) numerical equilibrium codes such as EFIT or variational moments equilibrium code (VMEC). For the rapidly evolving optimization studies that deliver very high-resolution results from investigations of plasmas with increasing problem size on the most powerful supercomputing systems, the practical choice—as exemplified by the *codesign* GTC-P code—is the category (1) analytically based equilibria. On the other hand, comprehensive production runs carried out by the *flagship* GTC code demand interfacing with the numerical equilibria of category, (2) that properly represent the actual experimental conditions.

The most accurate representation of the equilibrium in tokamaks is by using magnetic flux coordinates rather than Cartesian coordinates. This is because that most important equilibrium quantities, such as plasma temperature and density, can be shown to depend on the magnetic flux only. The flagship GTC code employs magnetic flux coordinates (ψ,θ,ζ) to represent the electromagnetic fields and plasma profiles, where ψ is the poloidal magnetic flux, θ is the poloidal angle, and ζ is the toroidal angle. Specifically, the inputs come from the numerical magnetic equilibrium and plasma profiles are obtained from EFIT/VMEC by transforming the equilibrium quantities defined in the cylindrical coordinates (R,ϕ,Z) to those defined in the magnetic coordinates (ψ,θ,ζ). The equilibrium data are provided by MHD equilibrium codes for the magnetic field strength B, and cylindrical coordinates (R,Φ,Z) of points forming magnetic flux surfaces. Additionally, the flux functions representing poloidal $g(\psi)$ and toroidal $I(\psi)$ currents, magnetic safety factor $q(\psi)$, and minor radius $r(\psi)$—defined as a distance from the magnetic axis along the outer mid-plane—are provided. First-order continuous B-splines are implemented for the one-dimensional (1D), two-dimensional (2D), and 3D functions to interpolate the complicated magnetic geometry and plasma profiles, which provide a good compromise between high numerical confidence and reasonable computational efficiency.

The GTC capability to carry out simulations of problems with general toroidal geometry has recently been extended to also include nonaxisymmetric configurations. For nonaxisymmetric devices, the equilibrium data are presented on the uniform (ψ,θ) grid for all $n = (1, 2, \dots N)$ toroidal harmonics. To reduce the computational load and memory usage, the transformation of nonaxisymmetric variables into spline functions of ζ is chosen for implementation in GTC, with spline coefficients associated with a particular grid point ζ_i being stored by processors with corresponding toroidal rank using message passing interface (MPI) parallelization. An example of GTC results for a stellarator plasma is shown in Figure 23.1 [13].

The GTC-P code deploys the so-called *large aspect ratio equilibrium*, which is an analytical model describing a simplified toroidal magnetic field with a circular cross-section. The associated model takes into account the key geometric and physics properties needed to carry out a meaningful study of the influence of increasing plasma size on magnetically confined fusion plasmas. Such an approach enables working with a sufficiently straightforward but nevertheless discovery-science-capable physics [3,4] code that makes more tractable the formidable task of developing the algorithmic advances needed to take advantage of the rapidly evolving modern platforms featuring, for example, both homogenous and hybrid architectures. The associated physics approach is to deploy GTC-P plasma size-scaling studies because it is a fast streamlined modern code with the capability to efficiently carry out computations at extreme scales with unprecedented resolution and speed on present-day multipetaflop computers [5]. The corresponding scientific goal is to accelerate progress toward capturing new physics insights into the key question of how turbulent transport and associated confinement characteristics scale from present generation laboratory plasmas to the much larger ITER-scale burning plasmas. This includes a systematic characterization of the spectral properties of the turbulent plasma as the confinement scaling evolves from a *Bohm-like* trend where the confinement degrades with increasing system size to a *Gyro-Bohm-like* trend where the confinement basically *plateaus* exhibiting no further confinement degradation as the system size further increases. *Lessons learned* achieved in a timely way from this codesign effort can be expected to expedite associated advances in the flagship GTC code in particular as well as to generally providing valuable information on PIC performance modeling advances to ongoing and future efforts in improving PIC code deployment on multipetaflop supercomputers on the path to exascale and beyond.

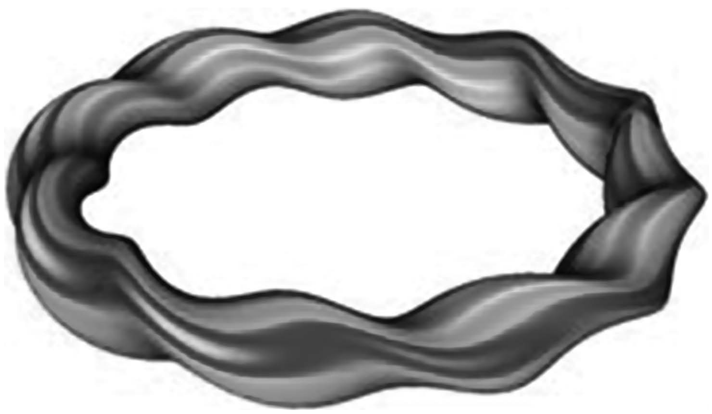
FIGURE 23.1 Illustration of GTC 3D mode structure for the $n = 1$ global Alfvén eigenmode in the large helical device (LHD) stellarator. (From D.A. Spong, *Phys. Plasmas* 22, 055602, 2015.)

23.2.5 Global PIC Grid Considerations

In accurately tracking the key physics in magnetically confined toroidal plasmas, the GTC and GTC-P codes utilize a highly specialized grid that follows the magnetic field lines as they twist around the torus (see Figure 23.1). This allows the code to retain the same accuracy while using fewer toroidal planes than a regular, non–field-aligned grid. From relevant physics considerations, because short wavelength waves parallel to the magnetic field are suppressed by Landau damping, increasing the grid resolution in the toroidal dimension will leave the results essentially unchanged. Consequently, a typical production simulation run usually consists of a constant number of poloidal planes (e.g., 32 or 64) wrapped around the torus. Each poloidal plane is represented by an unstructured grid, where the grid sizes in the radial and poloidal dimensions correspond approximately to the size of the gyroradius of the particles. As we consider larger plasma sizes (e.g., 2× in major and minor radius), the number of grid points in each 2D plane increases 4×. The number of grid points for a 3D grid increases 4× as well because the number of planes in the toroidal dimension remains the same for all problem sizes. For a modest-sized fusion device (e.g., the DIII-D tokamak at General Atomics in San Diego, CA), the associated plasma simulation typically uses ∼128,000 grid points in a 2D plane. As we move to the larger JET device and then eventually to the ITER size plasmas, the number of grid points increases 4× and 16×, respectively. Using a fixed number of 64 toroidal planes, the total number of grid points for an ITER-sized plasma will be ∼131 million. With 100 particles per cell resolution, an ITER-sized simulation will accordingly involve ∼13 billion particles. Tracking the dynamics of this large number of particles would of course be an extremely daunting task without access to leadership-class supercomputers.

23.3 ALGORITHMIC DETAILS

The basic PIC method is of course a well-established computational approach that simulates the behavior of charged particles interacting with each other through pair-wise electromagnetic forces. At each time step, the particle properties are updated according to these calculated forces. For applications on powerful modern supercomputers with deep cache hierarchy, a pure particle method is very efficient with respect to locality and arithmetic intensity (compute bound). Unfortunately, the $O(N^2)$ complexity makes a particle method impractical for plasma simulations using millions of particles per process.

Rather than calculating $O(N^2)$ forces, the PIC method, which was introduced by J. Dawson and N. Birdsall in 1968, employs a grid as the media to calculate the long range electromagnetic forces. This reduces the complexity from $O(N^2)$ to $O(N+M\log M)$, where M is the number of grid points and is usually much smaller than N. Specifically, the PIC simulations are being carried out using *macro* particles ($\sim 10^3$ times the radius of a real charged ion particle) with characteristic properties, including position, velocity, and weight. However, achieving high parallel and architectural efficiency is very challenging for a PIC method due to potential fine-grained data hazards, irregular data access, and low arithmetic intensity. The issue gets more severe as the high performance computing (HPC) community moves into the future to address even more radical changes in computer architectures as the multicore and manycore revolution progresses.

In this chapter, the computational approach involves the advanced development of a comprehensive *ab initio* PIC global (3D) code GTC and its codesign partner GTC-P, which cover equations underlying gyrokinetic theory. As highly scalable PIC codes used for studying microturbulent transport in tokamaks, GTC and GTC-P solve the gyro-phase-averaged Vlasov–Poisson set of equations (*gyrokinetic equations*) using discrete, charged particles. Particles interact with each other through a self-consistent field evaluated on a grid that covers the whole simulation domain. The charge of each particle is deposited on the grid by interpolating to its nearest grid points, resulting in a charge density that is then used in the evaluation of the field by solving the Poisson field equations. At the position of each particle, the field is then evaluated, again by interpolation, and used in the equations of motion to advance the particles.

The parallel algorithms in GTC-P are implemented with MPI and OpenMP. The original implementation included a 1D domain decomposition and a particle decomposition. This design had shown nearly perfect scaling with the number of particles. However, as the grid size is increased when simulating large fusion devices, the 1D domain decomposition produces a significant memory footprint. To address this issue, an extra dimension domain decomposition feature was added to GTC-P [15,24]. As a result, the particles are now fully distributed across all processes while the grid is split with the implementation of an appropriate 2D domain decomposition scheme—thereby greatly reducing the memory footprint and improving cache reuse. This algorithm was developed specifically for the Blue Gene systems in order to handle the limited amount of memory per node. It is a capability introduced in GTC-P to simulate very large fusion devices on BG/P with unprecedented efficiency—a key feature that was exercised in current studies to greatly facilitate examining the key question of how plasma microturbulence properties might be affected as the plasma size increases from that of existing experiments to the future very large plasmas characteristic of ITER. Finally, the GTC-P code has data parallelism at the loop level through the use of OpenMP directives. All the loops over the particles and grid points are fairly large and contain data parallelism. This method has been used very successfully on the multicore processors and has contributed to the excellent scaling of the GTC-P code on large-scale homogenous architecture supercomputers including the BG/Q *Mira* and *Sequoia* systems in the United States and on the Fujitsu K Computer in Japan (Figures 23.2 and 23.3).

Using resources from INCITE, previous early Science Projects (ESP) at the Argonne Leadership Computing Facility (ALCF), and Director's Discretionary allocations from both the ALCF and OLCF in the past few years, GTC-P has demonstrated excellent scalability to more than 100,000 cores on leadership computing facilities at Argonne National Laboratory (ANL) and Oak Ridge National Laboratory (ORNL). It has been successfully deployed for major scientific production runs on the IBM BG/Q/*Mira*—where the excellent weak scaling performance was carried over to much larger scale on LLNL's more powerful Sequoia system. These results are illustrated in Figure 23.2. In addition, it is relevant to note that the GTC-P code was the featured U.S. code in the G8 international exascale project in nuclear fusion energy (NuFuSE) (http://www.nu-fuse.com/) that was supported in the United States by the National Science Foundation (NSF) [14]. The G8 program helped provide

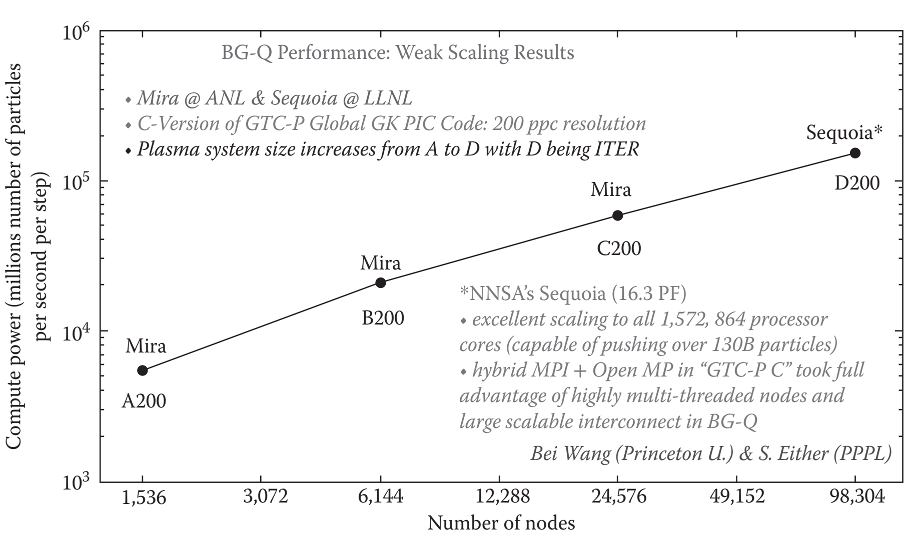

FIGURE 23.2 GTC-P code performance on world-class IBM BG/Q systems.

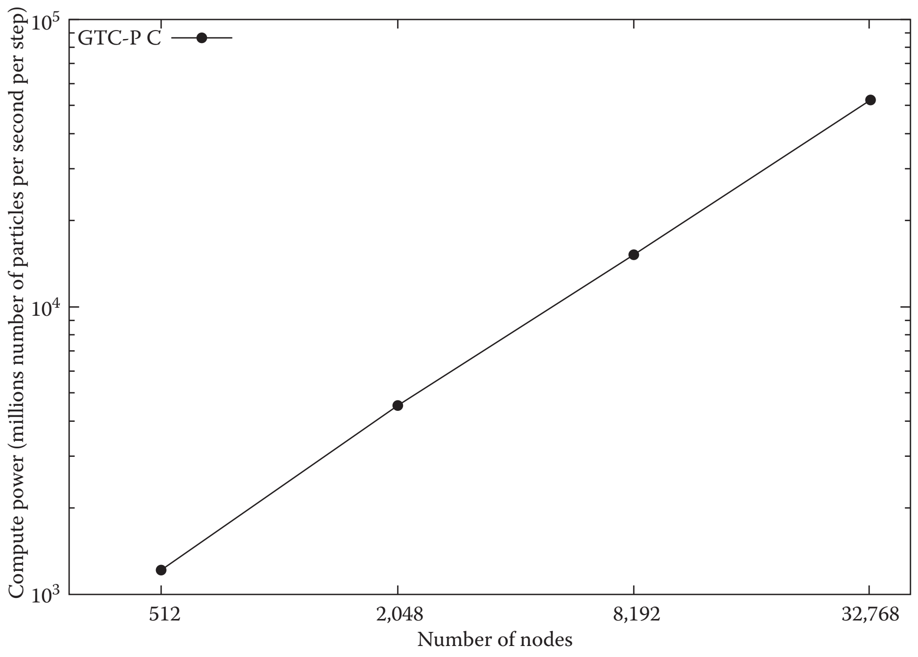

FIGURE 23.3 Excellent weak scaling of the GTC-P code achieved on the Fujitsu-K Computer in Japan.

unique access to a variety of international leadership class computational facilities such as the Fujitsu K Computer in Japan. As seen from subsequent stimulating new results [5], substantive impact can be expected to help stimulate progress in preparing for actual research engagement on ITER. In order to do so in a timely way, it is critically important that new software for extreme concurrency systems that demand increasing data locality be developed to help accelerate progress toward the ultimate goal of computational fusion research—a predictive simulation capability that is properly validated against experiments in regimes relevant for practical fusion energy production.

23.4 PROGRAMMING APPROACH

The basic parallel programming approach for global PIC codes such as GTC and GTC-P include (1) explicit message passing using MPI; (2) quasishared memory models such as Global Arrays for internode communication; (3) architecture-specific models such as CUDA for computing on graphic processing units (GPUs) and (4) directive-based compiler options such as OpenMP and OpenACC with possible promise of being more cross-machine portable between architectures.

In the course of describing global PIC code characteristics/considerations with respect to scalability, performance, portability, modern computational platforms, and external libraries, associated discussions will touch on the rationale for the chosen programming approach, and the associated balance between performance and portability. In particular, attention will be focused on specific challenges for global PIC applications in achieving efficiency on exascale architectures.

23.4.1 Scalability

This section focuses on a description of efforts to improve the scalability of the global PIC codes—well represented by GTC and GTC-P. In particular, key topics highlighted include (1) on-node thread scaling and (2) between node scaling.

The GTC/GTC-P codes have been designed with four levels of parallelism: (1) an internode distributed memory domain decomposition via MPI; (2) an internode distributed memory particle decomposition via MPI, (3) an intranode shared memory work partition implemented with OpenMP; and (4) a single instruction, multiple data (SIMD) vectorization within each core. This approach was shown to lead to nearly perfect scaling with respect to the number of particles [18].

In order to efficiently address large grid sizes and the associated significant memory increase, the domain decomposition in GTC-P is further extended in the radial dimension (beyond the toroidal dimension). This leads to a 2D domain decomposition and enables carrying out true weak scaling studies, where both particle and grid work are appropriately scaled. The multilevel particle and domain decompositions provide significant flexibility in distributed-memory task creation and layout. Although the ranks in the toroidal dimension are usually fixed as 32 or 64 due to Landau damping physics, there is freedom to choose any combination of process partitioning along the radial and particle dimensions. For scaling with a fixed problem size, the procedure involves first partitioning along the radial direction and then switching to particle decomposition for additional scalability. The decompositions were implemented with three individual communicators in MPI (toroidal, radial, and particle communicator), and further tuning is made available via options to change the order of MPI rank placement.

A gyrokinetic PIC simulation typically has highly anisotropic behavior, with the velocity parallel to the magnetic field being an order of magnitude larger than that in the perpendicular direction. Consequently, the message sizes in the toroidal dimension can be 10 times larger than those in the radial dimension at each time step. On Blue Gene systems with explicit process mapping, it is convenient and effective to group processes to favor the MPI communicator in the toroidal dimension [15]. For other systems, assigning consecutive ranks for processes within each toroidal communicator generally leads to improved performance.

To maximize on-node performance and efficiency, modern processor architectures have evolved with more cores and wider vector units in a single node. In order to fully exploit the emerging architectures on the path to exascale, it is important that application scientists design their software such that the algorithms and the implementations map well on the hardware for maximum scalability. In GTC/GTC-P, multicore parallelism is further exploited using shared-memory multithreading, which provides an additional multiplicative source of speed up. In an earlier version of GTC-P, *holes* were used to represent nonphysical *invalid* particles, that is, in a distributed environment, at every time step, the particles that are being moved to other processes are marked as *holes* and considered to be *invalid* in the local particle array. These invalid particles are then removed from the array periodically to empty memory space for new incoming particles. In this type of implementation, two particles in consecutive memory locations may have different operations in *charge* and *push* depending on if they belong to the same type of particles (valid or invalid) or not. This accordingly introduces difficulty for automatic vectorization. To maximize the usage of vector units, the latest version of GTC-P and GTC removes the holes completely for *charge* and *push* by filling the holes at the end of *shift* and using the new incoming particles sent from neighboring processors at every time step. If a process has sent more particles than received, then the remaining holes are filled with the last particles in the array. A similar strategy has been applied for the GPU implementation to remove the branch statement caused by the *holes*.

23.4.2 Performance

Global PIC code performance considerations involve a proper description of the GTC/GTC-P approach to achieving high performance on advanced architectures with focus on (1) improving data locality and vectorization; (2) improving thread scalability; and (3) making appropriate algorithmic changes.

Particle in cell algorithms is challenging to optimize on modern computer architectures due to issues such as data conflict and data locality. In GTC and GTC-P, parallel binning algorithms have been developed to improve data locality for *charge* and *push*. More specifically, several choices are provided to bin the particles, that is, along the radial dimension and along the poloidal dimension. The best binning strategy will be used for production runs by first running a few benchmarks. In GTC-P, the additional use of intrinsics has helped improve the vectorization of the binning implementation. On GPUs, the CUDA version of the binning algorithm was implemented using the Thrust Library.

To address the data conflict issue in *charge*, optimization strategies have been explored via static replication of grid segments that are coupled with synchronization via atomics, where the size of the replica may be traded for increased performance [16]. The best performance is often obtained by employing the full poloidal grid for each OpenMP thread. In GTC with only toroidal domain decomposition, the full poloidal grid replication dramatically increases the temporary grid-related storage for large size grid on manycore architectures such as the Intel Xeon Phi systems. As such, static replication of grid segments that are coupled with synchronization via atomics will likely be the best strategy. In GTC-P, the radial domain decomposition solves locality and memory pressure without resorting to costly atomics. In essence, because only a small segment of the full poloidal grid is required for a hazard-free charge deposition, the private grid replication strategy can be readily employed on a per thread basis for the best performance.

23.4.3 Portability

Global PIC codes such as GTC and GTC-P have demonstrated increasing capability for portability over the past few years across different architectures. In this section, the associated techniques applied for doing so are discussed along with examples of success achieved. In general, a high priority is being placed on portability in HPC because of the significant differences between quite different main-line approaches receiving heavy emphasis and by government investments—a prominent example being the major architectural differences between the upcoming 200 PF systems: the SUMMIT system at the OLCF and the AURORA system at the ALCF. Because both approaches have significant exciting potential for enabling accelerated performance at scale, most advanced applications—including prominent global PIC codes such as GTC/GTC-P—will continue to focus attention on achieving BOTH performance enhancement and portability. For example, performance portability of these advanced codes helps ensure—in a risk mitigation sense—the capability to perform very well on whichever platform proves to provide the greater eventual computing at extreme scale advantage.

GTC-P has been particularly successful in porting modern optimized versions across a wide range of multi petaflop platforms at full or near-to-full capability. Benefit is associated in part from the fact that GTC-P is not critically dependent on any third-party libraries. For example, this effort was initiated with the implementation a highly optimized Poisson solver with multithreading capability. Additional performance enhancement for both GTC and GTC-P has been obtained by utilizing a specialized damped Jacobi iterative solver [17]. In this iterative solver, the damping parameter was carefully chosen to favor the desired range of wavelengths for the fastest growing modes in plasma turbulence simulations. As a result, a small and fixed iteration count is sufficient to achieve the desired accuracy.

Although achieving the best performance on each explored architecture requires platform-specific optimization strategies, a *pluggable* software component approach in architecting the GTC/GTC-P application codes has proven to be a quite successful approach. Specifically, the interface is preserved across all implementations targeting CPU-based codes as well as GPU (or Xeon Phi) hybrid implementations. Components are chosen based on the target platform during the application build process. This enables having a unified code base with the best-possible performance, without sacrificing portability. Behind the unified interface, platform-specific optimization strategies are systematically investigated.

Some optimizations, such as sorting particles and vectorization, are common to all platforms, but implementation details differ. Other optimizations, such as handling non-uniform memory access (NUMA) issues and load imbalance, are specific to certain platforms. Designing routine interfaces is of crucial importance to allow portability without compromising performance-tuning opportunities.

GTC-P uses the MPI-3 standard for distributed-memory communication, including the exploration of explore one-sided communication. The motivation here is again to provide better portability for diverse architectures and programming models.

Significant advances in GTC-P on manycore processors with respect to portability and scalability have been recently achieved by porting the code to GPU systems with OpenACC 2.0 as a viable option instead of CUDA. This has led to the very recent success in porting and optimizing an OpenACC 2.0 version of GTC-P on the Sunway TaihuLight Supercomputer—the new No. 1 system on the international Top500 as of June 2016. The only approach for achieving good performance on TaihuLight requires software compatibility with their SWACC compiler—a customized OpenACC 2.0 syntax supported software.

In more generally considering the question of portability onto a broad variety of modern computational platforms, experiences with GTC-P indicate that machines such as the IBM BG/Q Mira demand at least 49,152-way MPI parallelism and up to 3 million-way thread-level parallelism in order to fully utilize the system. Although distributing particles to at least 49,152 processes is straightforward, the distribution of a 3D torus-shape grid among those processes is certainly a nontrivial task. For example, first considering the 3D torus as being decomposed into subdomains of uniform volume, the subdomains close to the edge of the simplest circular geometry system will contain more grid points than the core. This leads to potential load imbalance issues for the associated grid-based work.

Through a close collaboration with the Future Technologies Group at the Lawrence Berkeley National Laboratory, a new version of GTC-P has been developed and optimized to address the challenges in the PIC method for leadership-class systems in the multicore/manycore regime [16,22, 23]. As noted earlier in this document, GTC-P includes multiple levels of parallelism, a 2D domain decomposition, a particle decomposition, and a loop level parallelism implemented with OpenMP—all of which help enable this modern global PIC code to efficiently scale to the full capability of the largest extreme scale homogeneous HPC systems currently available [5]. Special attention has been paid to the load imbalance issue associated with domain decomposition [15]. To improve single node performance, a *structure-of-arrays* (SOA) data layout has been chosen for particle data. This is accompanied by aligning memory allocation to facilitate SIMD intrinsic binning of particles to improve locality and the use of *loop fusion* to improve arithmetic intensity. Irregular nested loops have been manually flattened to expose more parallelization to OpenMP threads. GTC-P features a 2D topology for point-to-point communication. On the IBM BG/Q system with 5D torus network, communications have been optimized with customized process mapping. Data parallelism has also been continuously exploited through SIMD intrinsics (e.g., QPX intrinsics on IBM BG/Q) and by improving data movement through software prefetching.

Overall, GTC-P has incorporated four levels of parallelism including (1) an internode distributed memory 2D domain decomposition via MPI, (2) an internode distributed memory particle decomposition via MPI, (3) an intranode shared memory work partition implemented with OpenMP, and (4) an SIMD vectorization within each core. In common with the large majority of codes in the fusion energy science/plasma physics application domain, GTC-P was originally written in Fortran language. However, to better facilitate interdisciplinary collaborations with computer science and applied math colleagues, modern versions of this code have been developed in C language as well as a CUDA implementation for dealing with GPUs. As just noted, this capability has recently been further advanced with the development and implementation of an OpenACC 2.0 version of GTC-P. Although the original Fortran version of this code is still used for verification purposes in cross-checking and benchmarking results, the primary utilization has involved the C and CUDA versions for performance studies and physics production runs on supercomputing systems such as the ALCF's *Mira* and the OLCF's *Titan*.

In dealing with heterogenous supercomputing platforms such as *Titan*, the approach followed in the deployment of global PIC codes involves off-loading the computationally intensive and highly scalable subroutines to GPUs, whereas the communication-dominant subroutines remain on CPUs. Performance, however, is known to be impeded due to the synchronization of atomic operations and the unavoidable memory transpose associated with the structure-of-array to array-of-structure data layout. To address this issue, the time-consuming global memory atomic operations have been replaced with local shared memory atomic operation. This R&D activity falls generally in the category of advances and challenges involving heterogeneous architectures.

23.4.4 External Libraries

A multilevel parallelization using MPI/OpenMP has been designed in GTC to scale up to millions of cores and to take advantage of the memory hierarchy of current generation parallel supercomputers. GTC is the first fusion code to reach the tera-scale in 2001 on the Seaborg computer at the National Energy Research Scientific Computing Center (NERSC) [3] and the petascale in 2008 on the Jaguar computer at ORNL in production simulations [19]. Through collaborations with computer scientists from hardware vendors, GTC was the first large-scale fusion code to fully utilize the heterogeneous architectures using GPU accelerators on the Tianhe-1A [20] and Titan, and using Intel Many Integrated Core (MIC) accelerators on Tianhe-2 [21]. As the codesign partner to the electromagnetic GTC code, the electrostatic GTC-P code has demonstrated its high-resolution portability capabilities on the top seven supercomputers worldwide [15]. These advances were enabled in significant measure by a well-established collaboration between Princeton University's Institute for Computational Science and Engineering (PICSciE) members Bei Wang and W. Tang and SciDAC SUPER Institute members L. Oliker and S. Williams and their colleagues in LBNL's Future Technology Group.

The flagship GTC code was originally written in Fortran 90. The CPU version has been parallelized using an MPI/OpenMP hybrid programming model with GPU and Intel Xeon Phi acceleration. Fortran 90 modules are used in this code to manage global data—with every class of global data (e.g., field data, particle data) having its own module. GTC has previously relied upon the Department of Energy (DOE)-funded PETSc toolkit at ANL to implement the electromagnetic parallel solvers. Advanced third-party packages, such as LLNL's HYPRE multigrid solver, have more recently been implemented in GTC as part of the current CAAR GTC Project at the OLCF. HYPRE can be used via the PETSc framework with a simple parameter change in a runtime configuration file.

23.5 SOFTWARE PRACTICES

The GTC code, which was originally written in Fortran 90, has a CPU version, which has been parallelized using an MPI/OpenMP hybrid programming approach with GPU and Intel Xeon Phi acceleration. This code uses Fortran 90 modules to manage global data—with every class of global data (e.g., field data, particle data) having its own module. GTC currently uses the DOE-funded PETSc toolkit to implement and utilize the electromagnetic parallel solvers. Advanced third-party packages, such as LLNL's HYPRE multigrid solver have recently been implemented in GTC and can be used via the PETSc framework with a simple parameter change in a runtime configuration file. The GTC features multiple levels of parallelism:

1. First, a 1D domain decomposition is implemented in the symmetric toroidal direction using MPI. The particles are divided between MPI processes in the domain decomposition

wherein each process owns a fraction of the total particles in that domain as well as a private copy of the local toroidal grid.

2. In order to further increase MPI parallelism, a second level of decomposition, a particle decomposition, is introduced. Particles are divided, but fields are shared between MPI processes in the particle decomposition. Field solvers are parallelized using all MPI processes in the particle decomposition.
3. The third level of parallelism is an intranode shared memory partitioning (via OpenMP) of both particle and grid-related computation. This results in a near-perfect scaling with the number of particles.

Moving forward, advanced radial domain decomposition methodology developed and successfully deployed in the codesign GTC-P code to efficiently reduce the memory footprint in larger problem size challenges is introduced into the flagship GTC code.

23.6 BENCHMARKING RESULTS

In this section, a description is provided of performance and portability of GTC and GTC-P, using the porting approach performed on these codes using the described programming approach. The discussion includes the following: (1) What has worked well and what has not; (2) the level of effort that was needed to do the refactoring and porting—with associated description of the benchmarking and profiling results; (3) illustration of parallel scaling behavior, displayed on a logarithmic scale such that linear or ideal scaling is shown as a straight line for either weak or strong scaling; and (4) a description of *time to solution*, which is a metric more meaningful than parallel efficiency, as the latter is often based on an arbitrary data point. Of course, *time to solution* is the key metric that counts the most for computational domain scientists. Finally, while exascale resources are, obviously, not as yet available, the topic of extrapolation to exascale resources, based on expected architectural roadmaps, is be discussed.

Two sets of weak scaling studies for the comprehensive GTC code have been carried out on Titan up to nearly the full system 16,384 nodes. However, at the time of this study, a significant number of the Titan nodes were unavailable, making it impossible to run on all 18,688 nodes. The first test set is called *particle weak scaling study*, where the grid size is held fixed, but the total number of particles is scaled up. The second set of test is called *hybrid weak scaling study*, where both the grid size and the total number of particles are scaled. The first study holds the number of particles per MPI rank and the number of grid cells per MPI rank nearly constant, thus representing a conventional weak scaling study. However, the second study is a more realistic performance scaling study based on a typical production run of the code: grid size is proportional to the square root of number of nodes. For both sets of weak scaling study, the number of particles per processor is fixed at 3.2 million. Compared with CPU (16 cores AMD 6274), GPU (NVIDIA K20×) has boosted the overall performance by $1.6 - 3.0\times$. The decrease of the performance speedup in large processor counts is mainly due to the increased portion of non-GPU accelerated subroutines as well as MPI time (Figures 23.4 and 23.5).

The GTC Poisson solver currently runs on the CPU. Though it is presently not the most time-consuming part of GTC simulations, the solver time requirements have become more significant since other parts of the code have been accelerated using GPUs. The standard PETSc solver has accordingly been replaced with a HYPRE multigrid solver as part of the CAAR GTC Project. This solver has the clear advantage of being threaded to effectively use the CPUs while also being scalable to many compute nodes. Figures 23.6 and 23.7 show comparative timings of the PETSc solver and the HYPRE multigrid solver for a representative set of GTC test cases. The HYPRE solver for these cases is ~4× faster than the standard PETSc solver and has better scaling properties.

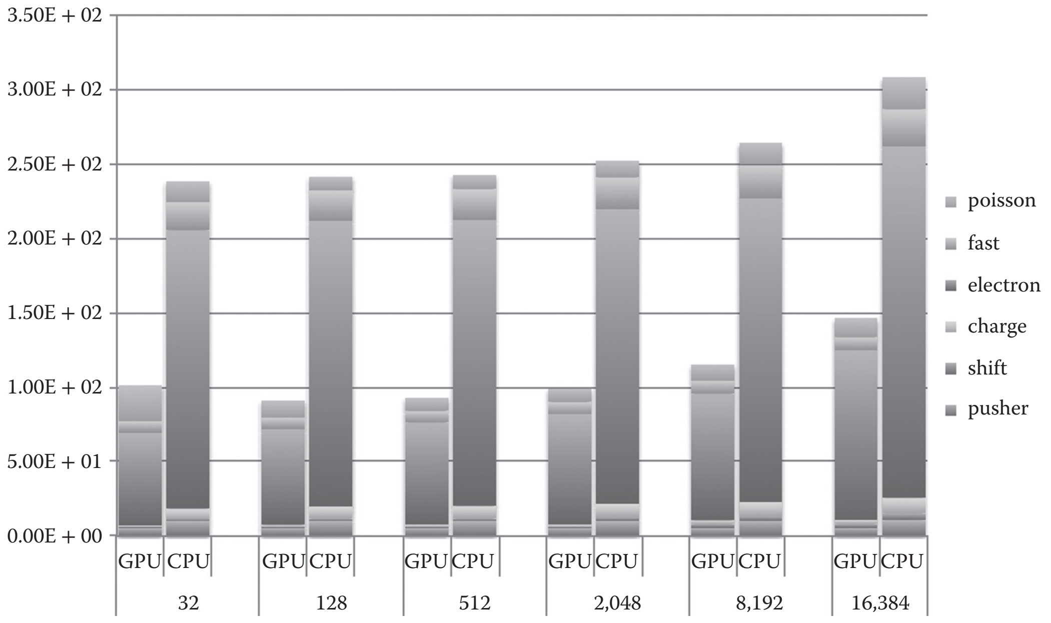

FIGURE 23.4 The timing breakdown for GTC particle weak scaling study on Titan. *Note*: *x*-axis is the number of nodes and *y*-axis the total wall-clock time. GPUs are shown to deliver up to 3× speedup compared with CPUs.

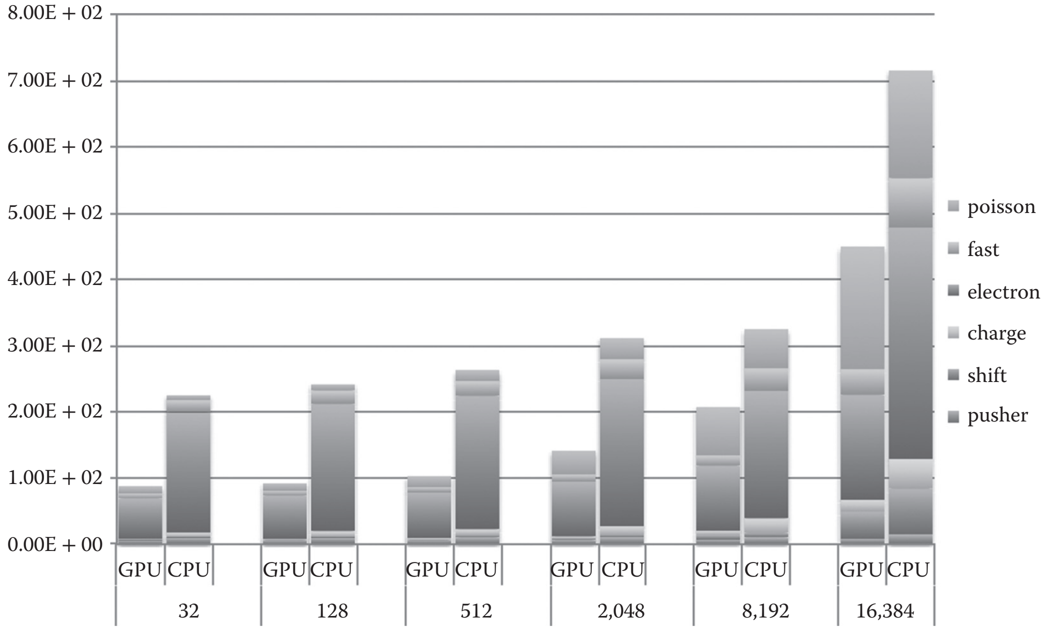

FIGURE 23.5 The timing breakdown for GTC hybrid weak scaling study on Titan. Here the work per processor is increased as node count is increased. *Note*: *x*-axis is the number of nodes and *y*-axis the total wall-clock time. GPU delivers up to 3.0× speedup compared with CPU.

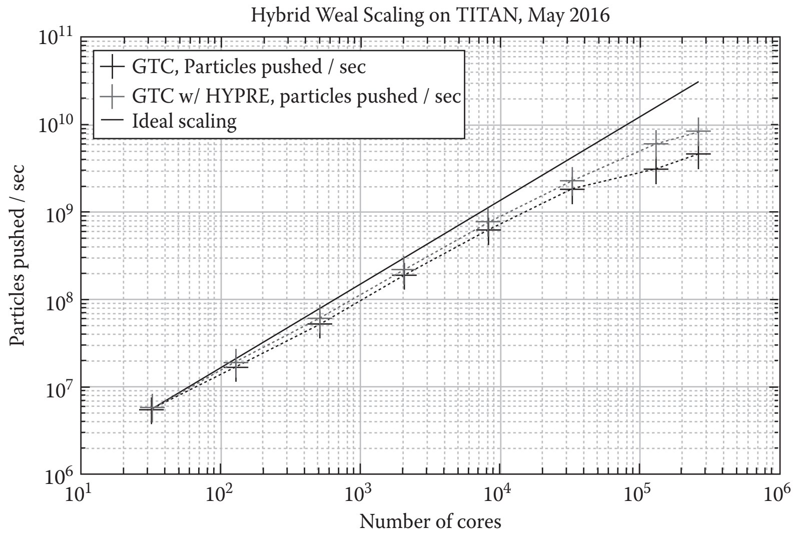

FIGURE 23.6 Comparative performance of the PETSc versus HYPRE solvers for representative GTC Code cases on Titan (May, 2016).

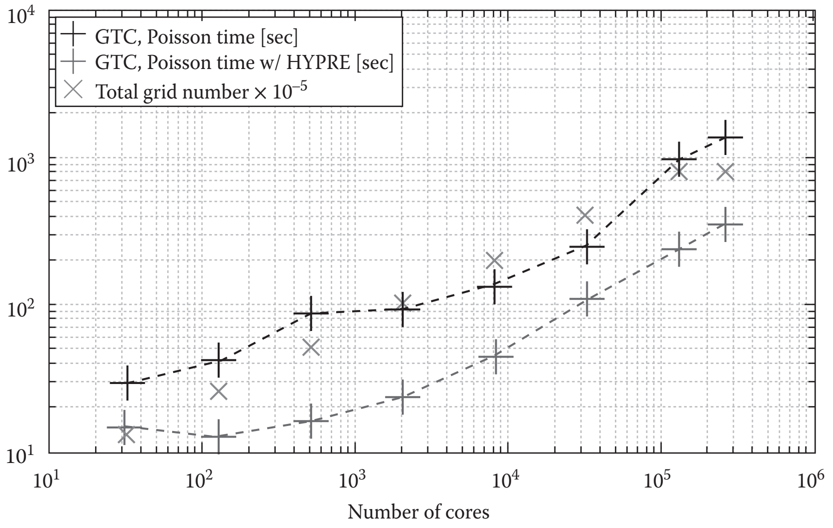

FIGURE 23.7 Weak scaling of GTC on Titan (top), with the number of nodes ranging from 32 to 16,384 (88% of the whole machine). Both grid number and total particle number are increased, but the number of particles per core remains constant. The Poisson time (bottom) shows the improved performance due to the new deployment of the HYPRE multigrid solver. Total grid number for this scaling study is also shown.

The benchmarking results associated with GTC-P—a highly scalable code developed to efficiently utilize modern parallel computer architectures at the petascale and beyond—features simulations of confinement physics for large-scale MFE plasmas that have been carried out for the first time with very high phase-space resolution and long temporal duration to deliver important new scientific results that are cross-benchmarked versus results achieved on various top supercomputing systems. This was enabled by deployment of this code on world-class *homogeneous* architecture systems such as the IBM BG/Q *Mira* at the ALCF and also *Sequoia* at LLNL and the Fujitsu K Computer at RIKEN, Kobe, Japan. Some specific accomplishments are summarized in Figures 23.2 and 23.3.

The success of these efforts was greatly facilitated by the fact that true interdisciplinary collaborative effort with Computer Science and Applied Math scientists has accelerated completion of the modern C as well as GPU-compatible versions of the GTC-P code. The demonstrated capability to run at scale on the largest open-science BG/Q system (*Mira* at the ALCF) opened the door to obtain access to the National Nuclear Security Administration's (NNSA) *Sequoia* system at LLNL, which then produced the outstanding weak scaling performance results shown on Figure 23.2.

With regard to a more global perspective, the G8-sponsored international program on computing at extreme scale [14], has enabled this project to gain collaborative access to a number of the top international supercomputing facilities—including the Fujitsu K Computer, previously Japan's top-ranked system worldwide. Results from weak-scaling studies carried out on the K-computer are illustrated in Figure 23.3.

Having demonstrated the ability to effectively utilize the most powerful *homogeneous* supercomputing platforms worldwide, GTC-P R&D efforts have also examined performance characteristic on *heterogeneous* architectures. More generally, these studies represent productive investigations of extreme scale science across advanced scientific computing basic research programs with fusion energy science as an illustrative application domain. Here, the focus was on developing new algorithms for advanced *heterogeneous* supercomputing systems such as the GPU/CPU *Titan* at DOE's OLCF and the Intel Xeon Phi/Intel Xeon *Stampede* at the National Science Foundation's (NSF), Texas Advanced Computing Center (TACC). In doing so, a new version of GTC-P code was developed that features algorithms, which include new heterogeneous capabilities for deployment on hybrid GPU (NVIDIA K20)/CPU as well as the Intel Xeon Phi/Intel Xeon systems such as Stampede and also TH-2 in China. From a verification perspective, this research effort includes systematic comparison of new results against the successful work described in earlier in studies that featured high-resolution, long temporal scale simulation results obtained on world-class *homogeneous* systems such as the IBM BG/Q *Mira* at the ALCF, *Sequoia* at LLNL, and the K-Computer in Kobe, Japan.

The development of a GPU version of GTC-P suitable for accelerator-based architectures powered by GPUs started with focus on off-loading the particle-based phases—charge deposition, particle push, and particle shift—to the GPUs, while keeping the grid-based phases of the code on the CPUs. Consequently, the GPU implementation includes three programming models: CUDA and OpenMP within a node and MPI between nodes. The particle-based phases are good candidates for GPU implementation because they are especially computation intensive, taking at least 80% of the total computational time of the code [15].

Exploiting massive fine-grained parallelism on GPUs is nevertheless a nontrivial challenge for PIC codes, which feature fine-grained data hazards, irregular data access, and low computational intensity. For example, memory locality that typically *improves* the performance of most routines actually *degrades* the performance for atomics because of access conflicts. These conflicting requirements for locality and conflict avoidance have made previous [16]—as well as continuing—efforts to optimize the performance of modern codes on GPU systems both interesting and challenging.

The particle shift phase of the code consists of four steps: (1) finding the particles on the GPUs that need to be sent to other MPI processes, (2) copying these particles from GPUs to CPUs through a PCI bus, (3) sending/receiving particles with MPI_Sendrecv on CPU, and (4) copying the received particles from CPUs to GPUs. Performance is found to be impeded in finding and buffering those particles on GPUs and in explicit memory transfers between multiple memory spaces within a compute node.

Several optimization strategies to boost the code performance on different GPU architectures have been developed. On the older Fermi chip, the performance degraded mainly due to slow global atomic operation. We have accordingly replaced the associated global memory operation with a shared memory atomic operation—but with more memory usage. On the newer generation Kepler chip, where the performance of global atomic operations is dramatically improved, we have kept the global memory access with less memory usage. In addition, we have optimized the data layout such that the access of the global memory is achieved in a more coalesced way. In addition, a binning-based shift algorithm is being developed to improve the performance in finding and buffering shifted particles on GPUs with massive parallelism. This is the first step of the particle shift phase of the particle-based operations.

More recent developments of advanced algorithms addressing the programming challenges on hybrid GPU/CPU systems have produced some quite encouraging results (enabled by OLCF Director's Discretionary time allocation) on Titan—a Cray XK7 system with 299,008 CPU cores and 18,688 K20 NVIDIA GPUs. Quite favorable particle-scaling performance results have been obtained for fixed plasma size with the starting point being the porting of a significantly improved GPU version of GTC-P to Titan. For example, excellent performance of the GPU-version of GTC-P has recently been demonstrated on the *Titan* system at the OLCF. The results of associated very favorable weak-scaling studies are illustrated in Figure 23.8.

In ongoing investigations of the comparative capabilities of powerful *heterogeneous* system versus those of the petascale *homogeneous* systems, it is of course necessary to move beyond particle scaling studies to carry out true weak-scaling studies, where the plasma size increases.

In the GPU implementation in the GTC-P code, the GPU and CPU operations were not overlapped. For example, when the GPU is busy with charge deposition, the CPU remains idle. To utilize the full node power, two different methods are now being developed in this code. One approach is to distribute the particles among CPUs and GPUs such that they work concurrently. The ratio of the number of particles on each device (CPU or GPU chip) is carefully selected such that the work on CPUs and GPUs are well balanced. This is also the strategy we are using on heterogeneous systems with MIC coprocessors (which will be described in more detail below). Another approach is to utilize the idle processors for in situ data analysis [25]. In view of the fact that for applications with great code complexity where the accelerator/coprocessor provides only a moderate speedup compared

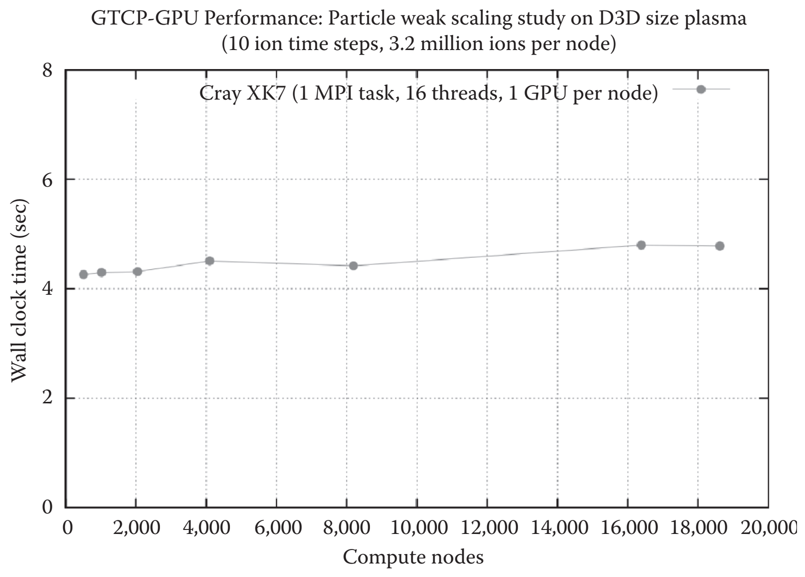

FIGURE 23.8 Particle weak scaling results obtained from the graphic processing unit (GPU)-version of the GTC-P code as carried out for a fixed D3D-size tokamak plasma on the Titan LCF at ORNL.

with commodity CPUs, both of the promising alternative approaches just noted here can better utilize CPUs together with the accelerator/coprocessor and thus significantly improve the performance in terms of *time to solution*.

In order to fully utilize the parallelism in the coprocessor/accelerator, it is important that algorithms be designed such that the code can exploit the SIMD vectorization. On GPUs, this translates to coalescent memory access and we accordingly structured the data array layout to facilitate SIMD vectorization. These developments can be expected to have a strong impact in dealing with the multi-threading challenges for efficient deployment of a large number of processors on modern GPU/CPU heterogeneous systems. Continuing engagement with key personnel at NVIDIA is a significant *codesign* asset in these R&D efforts.

With regard to the deployment experience on NSF's *Stampede* system with Intel Xeon Phi (*MIC*) coprocessors at the TACC, each coprocessor is considered at present as a separate node, and the code has been run in symmetric mode. In dealing with 2D domain decomposition, we distribute each subdomain along with its associated particles to a single *Stampede* server node. We further distribute the particles in one subdomain among 2 Intel Xeon E5 and Intel Xeon Phi coprocessor by turning on the particle decomposition. This means that each Intel Xeon E5 and Intel Xeon Phi in a node carries a fraction of the total number of particles in that subdomain. Having the host and the coprocessor in a node share the same subdomain (for particle decomposition only) can avoid running the grid-based subroutines on coprocessors redundantly.

Currently, the particles are divided evenly among the host and the coprocessor. Later, we will distribute the particles such that the work on the host and the coprocessor is well balanced. The chip level parallelism is exploited with OpenMP and SIMD vectorization. For example, on a node with 2 Intel Xeon E5, we use OpenMP with 16 threads. Similarly, on a coprocessor with the Intel Xeon Phi, we use OpenMP with 240 threads.

The data conflict issue in the charge deposition phase is addressed by providing each thread a private copy of the local grid following by a reduction operation to merge all copies together. The summation order is carefully arranged such that no costly synchronization is required. It should be noted here that the private copy strategy is applicable when the code has operational 2D domain decomposition needed to efficiently deal with large-size plasmas.

From the perspective of *codesign R&D*, it is appropriate to note here that ongoing discussions with lead designers at Intel can be expected to provide access to more advanced Intel MIC systems. In addition, we will have access to and be able to test our newly developed hybrid algorithms on the powerful heterogeneous TH-2 Intel MIC hybrid supercomputer in Guangzhou, China—currently the #2-rated system worldwide.

23.7 TIME-TO-SOLUTION AND ENERGY-TO-SOLUTION COMPARATIVE STUDIES

Energy is becoming an increasingly large impediment to advances in supercomputing. The net energy efficiency of large scientific simulations can be particularly nonintuitive as one moves from one processor or network architecture to the next as the interplay between performance and power is highly dependent on algorithm and architecture. Using power measured under actual load via system instrumentation, Table 23.2 shows the energy per time step on 4K nodes of Mira, Titan, and Piz Daint when using 80M grid points, 8B ions, and 8B electrons. We observe that although Mira required the most wall clock time per time step, it also required the least power per node. When combined, this ensured Mira required the least energy per time step of all platforms considered. Conversely, using the host-only configurations on Titan and Piz Daint required between 2× and 4× the energy with the difference largely attributable to the lack of scalability on Titan. Interestingly, although accelerating the code on

TABLE 23.2
***ENERGY-TO-SOLUTION* ESTIMATES (for Mira, Titan, and Piz Daint)**

	CPU-Only			CPU+GPU	
	Mira	**Titan**	**Piz Daint**	**Titan**	**Piz Daint**
Nodes	4,096	4,096	4,096	4,096	4,096
Power/node (W)	69.7	254.1	204.9	269.4	246.5
Time/step (s)	13.77	15.46	10.00	10.11	6.56
Energy (kWh)	1.09	4.47	2.33	3.10	1.84

- Energy per ion time step (kWh) by each system/platform for the weak-scaling, kinetic electron studies using 4 K nodes.
  (Watts/node) * (#nodes) * (seconds per step) * (1 kW/1,000 W) * (1 h/3,600 sec)
- Power/Energy estimates obtained from system instrumentation including compute nodes, network, blades, AC to DC conversion, etc.

these platforms significantly reduces wall clock time per time step, it only slightly increased power. As such the energy required for the GPU-accelerated systems was reduced nearly proportionally with run time. Optimizing for energy-to-solution could be at odds with time-to-solution when we use technologies such as dynamic voltage and frequency scaling (DVFS). When internode communication dominates the execution time by scaling down the CPU frequency we observed up to 25% reduction in energy consumption (for the C problem size) for an increase in the execution time by 28%. We conducted such an experiment on an Intel Haswell-based Cray XC40 system. Although supported by much hardware, DVFS control is not enabled on most of the systems studied. As such, we could not explore such optimization on all systems. Another impediment to adopting such technology is the policy adopted for resource allocation by HPC compute facilities, which is based on CPU-hours rather than energy consumption. This policy makes the most performant solution the best from the user perspective.

With regard to energy-efficient scientific computing, instrumenting scientific applications to measure energy when running on large supercomputing installations today can be cumbersome and obtrusive—requiring significant interaction with experts at each center. As such, most applications have little or no information on energy-to-solution across the architecture design space spectrum. In order to affect energy-efficient codesign of supercomputers, energy measurement must be always-on by default with, at a minimum, total energy and average power reported to the user at the end of an application. By reporting energy by component (memory, processor, network, storage, etc), scientists and vendors could codesign their applications and systems to avoid energy hotspots and produce extremely energy-efficient computing systems.

23.8 CONCLUDING COMMENTS

As a final comment, it is appropriate to note that a broader impact of the work presented in this chapter is the delivery of benefits to PIC codes in general because the associated codes share a common algorithmic foundation. For example, the continuing developments targeted in the current project can be expected to have a strong impact in dealing with the multithreading challenges for efficient deployment of a large number of processors on modern heterogeneous systems with coprocessors (GPU or MIC)—advances that should prove beneficial to any particle-mesh algorithm for such systems.

REFERENCES

1. W. Tang et al. *Scientific Grand Challenges: Fusion Energy Sciences and the Role of Computing at the Extreme Scale*, PNNL-19404, 212 pp. (March, 2009). http://www.er.doe.gov/ascr/ProgramDocuments/Docs/FusionReport.pdf.
2. R. Rosner et al. Opportunities & challenges of exascale computing—DoE Advanced Scientific Computing Advisory Committee Report, Office of Science, U.S. Department of Energy Fall (November, 2010).
3. Z. Lin et al. Size scaling of turbulent transport in magnetically confined plasmas, *Phys. Rev. Lett*. 88, 195004 (2002).
4. B. F. McMillan, et al. System size effects on gyrokinetic turbulence, *Phys. Rev. Lett*. 105, 155001 (2010).
5. W. Tang et al. Extreme scale plasma turbulence simulations on top supercomputers worldwide, *SC16 International Conference on High Performance Computing, Networking, Storage and Analysis*, Salt Lake City, UT; oral presentation, paper 399, 14–18 November, 2016.
6. Z. Lin et al. A fluid-kinetic hybrid electron model for electromagnetic simulations, *Phys. Plasmas* 8, 1447–1450 (2001).
7. Y. Chen and Parker, S. E. A δf particle method for gyrokinetic simulations with kinetic electrons and electromagnetic perturbations, *J. Comput. Phys.*, 189, 463–475 (2003).
8. E. Startsev and W. W. Lee. Finite beta simulations of microinstabilities, *Phys. Plasmas*, 21, 022505 (2014).
9. W.J. Deng et al. Linear properties of reversed shear Alfvén eigenmodes in the DIII-D tokamak, *Nucl. Fusion*, 52, 043006 (2012).
10. D. Liu et al. Verification of gyrokinetic particle simulation of current-driven instability in fusion plasmas. II. Resistive tearing mode, *Phys. Plasmas*, 21, 122520 (2014).
11. D. Liu et al. A finite-mass fluid electron simulation model for low-frequency electromagnetic waves in magnetized plasmas, *Plasma Phys. Control. Fusion*, 53, 062002 (2011).
12. D. Liu et al. Verification of gyrokinetic particle simulation of current-driven instability in fusion plasmas. III. Collisionless tearing mode, *Phys. Plasmas*, 23, 022502 (2016).
13. D. A. Spong. 3D toroidal physics: Testing the boundaries of symmetry breaking, *Phys. Plasmas*, 22, 055602 (2015).
14. G8 Research Council's "Exascale Computing for Global Scale Issues" Project in Fusion Energy "NuFuSE" [http://www.nu-fuse.com/]—An international HPC collaborative research project involving the U.S., France, Germany, Japan, and Russia (2011–2014). It is supported in the United States by the National Science Foundation (NSF)—with W. Tang as the US PI.
15. B. Wang. Kinetic turbulence simulations at extreme scale on leadership-class systems, oral presentation at SC13, published in SC13, *Proceedings of 2013 International Conference on High Performance Computing, Networking, Storage and Analysis,* Denver, CO. http://www.hpcwire.com/2013/11/16/sc13-research-highlight-extreme-scale-plasma-turbulence-simulation/ (November, 2013).
16. K. Madduri, et al. Memory-efficient optimization of gyrokinetic particle-to-grid interpolation for multicore processors, *Supercomputing*, (SC), 2009.
17. Z. Lin et al. Method for solving the gyrokinetic Poisson equation in general geometry, *Phys. Rev. E*, 52, 5646 (1995).
18. S. Ethier et. al. Gyrokinetic particle-in-cell simulations of plasma microturbulence on advanced computing platforms. *Journal of Physics: Conference Series*, 16, 1–15 (2005).
19. Y. Xiao and Lin, Z. Turbulent transport of trapped electron modes in collisionless plasmas, *Phys. Rev. Lett.*, 103, 085004 (2009).
20. X. Meng et al. Heterogeneous programming and optimization of gyrokinetic toroidal code and large-scale performance test on th-1a, *International Supercomputing Conference*, Leipzig, Germany, pp. 81–96. Springer (2013).
21. J. Dongarra. Visit to the National University for Defense Technology Changsha, China (2013). http://www.netlib.org/utk/people/JackDongarra/PAPERS/tianhe-2-dongarra-report.pdf.
22. K. Madduri et al. Gyrokinetic toroidal simulations on leading multi- and many-core HPC systems, *The International Conference for High Performance Computing, Networking, Storage, and Analysis (SC'11)*, Seattle, WA (2011).
23. K. Madduri et al. Gyrokinetic particle-in-cell optimization on emerging multi- and many-core platforms, *Parallel Comput.* 37(9), 501–520 (2011).

24. S. Ethier et al. Petascale parallelization of the gyrokinetic toroidal code. *Proceedings of 2010 High Performance Computing for Computational Science (VECPAR'10)*, Berkeley, CA, 2010.
25. F. Zheng et al. Gold Rush: Resource efficient *In Situ* scientific data analytics using fine-grained interference-aware execution. *Proceedings of 2013 International Conference on High Performance Computing*, *Networking, Storage and Analysis*, Denver, CO, 2013.

RESEARCH TEAM

- Princeton University: *Prof. William Tang, Dr. Bei Wang, Princeton Institute of Computational Science & Engineering (PICSciE), Dr. Stephane Ethier, Princeton Plasma Physics Laboratory (PPPL)*
- University of California, Irvine: *Prof. Zhihong Lin), Dr. Wenlu Zhang, Dr. Jian Bao, Dr. Animesh Kuley, Dept. of Physics & Astronomy*
- ETH Zurich: *Prof. Torsten Hoefler, Grzegorz Kwasniewski, Intel Parallel Computing Center*
- Lawrence Berkeley National Laboratory (LBNL): *Dr. Leonid Oliker, Dr. Samuel Williams, Dr. Khaled Ibrahim, Computer Science Department*
- Penn State University: *Prof. Kamesh Madduri, Department of Computer Science & Engineering*
- Argonne Leadership Computing Facility (ALCF): *Dr. Hal Finkel, & Dr.Venkat Vishwanath*
- Texas Advanced Computing Center (TACC): *Dr. Carlos Rosales-Fernandez*
- Oak Ridge National Laboratory: *Dr. Scott Klasky, Dr. Ed D'Azevedo, Dr. Wayne Joubert*
- NVIDIA, Inc.: *Dr. Peng Wang, Dr. Tom Gibbs*