Kinetic Turbulence Simulations at Extreme Scale on Leadership-Class Systems

Bei Wang[†], Stephane Ethier[‡], William Tang[†,‡],
Timothy Williams[♮], Khaled Z. Ibrahim[*], Kamesh Madduri[§,*], Samuel Williams[*], Leonid Oliker[*]
[†] *Princeton Institute of Computational Science and Engineering, Princeton University, Princeton, NJ, USA*
[‡] *Princeton Plasma Physics Laboratory, Princeton, NJ, USA*
[*] *Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA, USA*
[§] *Computer Science and Engineering, The Pennsylvania State University, University Park, PA, USA*
[♮] *Argonne Leadership Computing Facility, Argonne National Laboratory, Argonne, IL, USA*

ABSTRACT

Reliable predictive simulation capability addressing confinement properties in magnetically confined fusion plasmas is critically-important for ITER, a 20 billion dollar international burning plasma device under construction in France. The complex study of kinetic turbulence, which can severely limit the energy confinement and impact the economic viability of fusion systems, requires simulations at extreme scale for such an unprecedented device size. Our newly optimized, global, *ab initio* particle-in-cell code solving the nonlinear equations underlying gyrokinetic theory achieves excellent performance with respect to "time to solution" at the full capacity of the IBM Blue Gene/Q on 786,432 cores of *Mira* at ALCF and recently of the 1,572,864 cores of *Sequoia* at LLNL. Recent multithreading and domain decomposition optimizations in the new GTC-P code represent critically important software advances for modern, low memory per core systems by enabling routine simulations at unprecedented size (130 million grid points ITER-scale) and resolution (65 billion particles).

1. INTRODUCTION

In this study we describe the performance optimization on leadership-class systems of global gyro-kinetic particle-in-cell (PIC) codes used to simulate the complex kinetic dynamics governing key magnetic confinement properties of fusion-grade plasmas [24]. The fundamental scientific motivation for such research is that the fusion of light nuclides forms the basis of energy release in the universe and can potentially be harnessed and used as a clean and sustainable supply of energy on Earth. In order to build the scientific foundations needed to develop fusion energy, a key component is the timely development of high-fidelity predictive simulation capability for modern magnetically confined fusion plasmas especially with relevance to ITER [10], a 20 billion dollar international burning plasma device under construction in France and involving the partnership of seven governments representing over half of the world's population. An associated central physics challenge is understanding, predicting, and controlling instabilities caused by the unavoidable spatial variations (gradients) in such systems, i.e., the occurrence of turbulent fluctuations ("microturbulence"), which can significantly increase the transport rate of heat, particles, and momentum across the confining magnetic field. The practical mission importance associated with this scientific grand challenge is that microturbulence can severely limit the energy confinement time for a given machine size and therefore it's performance and economic viability. Understanding and possibly controlling the balance between these energy losses and the self-heating rates of the actual fusion reaction is key to achieving the efficiency needed to help ensure the practicality of future fusion power plants. Sufficiently realistic calculations of turbulent transport can only be achieved through advanced simulations.

For advanced kinetic simulations, the gyro-kinetic method is a well-established approach that evolves the distribution functions of the plasma particles in a five-dimensional phase space [13]. The rapid spiraling of the charged particles about magnetic field lines allows one of the dimensions to be neglected. Leading global PIC microturbulence codes solve the nonlinear integro-differential equations underlying gyrokinetic theory, employing modern numerical techniques and implementations. While the excellent scaling of global PIC codes on modern computing platforms is well established, significant challenges remain. In order to effectively address the outstanding open issues in fusion plasma physics such as the scaling of the energy confinement time with system size, access to computational resources at the multi-petascale level and beyond, as well as the development of associated algorithm and general software advances will clearly be required. Since it is expected that both the software and the hardware will have to be developed in a "co-design" sense in order to achieve progress in a timely manner, the research described in this paper deals with the demonstration that our advanced global PIC codes can deliver high-resolution simulation capabilities that are needed to deliver new scientific results at an accelerated pace by effective utilization of the powerful capabilities of the IBM Blue Gene/Q (BG/Q). This has involved our successful development of the radial domain decomposition capability in gyrokinetic toroidal code (GTC). The code with the key additional domain decomposition in radial dimension is named as GTC-

SC13 November 17-22, 2013, Denver, Colorado, USA

http://dx.doi.org/10.1145/2503210.2503258

P. The original version of GTC-P has demonstrated unprecedented efficiency for turbulent transport scaling analysis spanning the range from present generation experiments to the large ITER-scale plasmas [1, 4, 14]. However, earlier investigations [4] only explored flat MPI programming and suffered from load imbalance issues at high processor counts. In addition, the associated Poisson solver did not have the modern multi-threading capability needed to enable efficient performance on multi- and many-core architectures.

In order to more efficiently benefit from computer science advances in deploying multi-threading capabilities to facilitate large-scale simulations on modern low memory per core systems, a highly optimized version of GTC [19] is our starting point instead of the original version of GTC-P. We then proceed to develop and implement a highly efficient radial domain decomposition capability with special attention to load imbalance issues. In addition, OpenMP multi-threading is included in all subroutines of the new GTC-P code, a modern implementation with the capability to efficiently carry out computations at extreme scales with unprecedented resolution and speed on present-day multi-petaflop supercomputers. The associated impact on scientific advances is that the new GTC-P code now enables systematic investigations to acquire improved understanding of the important physics phenomenon, such as the favorable "Bohm to Gyro-Bohm" size scaling trend. This will enable 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.

The extreme-scale performance results of GTC-P presented in this paper are obtained from the BG/Q system, the third generation of the IBM Blue Gene supercomputer series. A true weak scaling study is carried out where both the grid size and the number of particles are increased in proportion to the number of processor cores. In contrast to previous numerical experiments that focused on relatively small-scale plasma simulations, this paper describes results for ultra-scale simulations with unprecedented phase-space resolution that are well demonstrated in ITER-size runs with up to 500 particles per cell. This corresponds to 130 million grid points and 65 billion particles in a true "path to extreme scale" advanced simulation. It should be noted that to enable portability across a variety of supercomputing platforms, the new GTC-P code does not depend on any third party libraries. To demonstrate this flexibility, scaling results have been obtained for two leading CRAY systems, the Cray XE6 system (Hopper) and the Cray XC30 (Edison). In addition, although not discussed in this paper, GPU algorithmic advances have been successfully developed and implemented in the GTC-GPU code [9, 19], and future work will aggressively target these methods to effectively leverage extreme-scale GPU-accelerated computing platforms, such as Titan at the Oak Ridge Leadership Class Facility. Overall, the delivery of otherwise unachievable scientific results that demand efficient use of leadership computing platforms at tens of petaflops in the ways described in the present paper can be viewed as an appropriate metric for the path to extreme-scale grand challenge projects.

Finally, it is important to highlight that GTC-P is the featured U.S. code in the currently funded G8 international exascale project "NuFuSE" [23]. Associated scientific findings can be expected to have substantive impact on accelerating progress in DoE's Office of Science research engagement in ITER, as well as in the international G8 Exascale pilot project in Nuclear Fusion Energy (NuFuSE) that is supported in the U.S. by National Science Foundation.

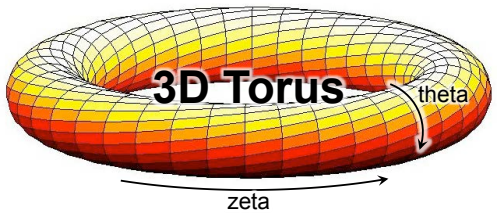

Figure 1: An illustration of GTC's 3D toroidal grid.

2. GYROKINETIC TOROIDAL MODEL

The study of low-frequency microturbulence in high-temperature, magnetically confined plasmas requires the use of the kinetic model described by the "Vlasov" equation in six-dimensional phase space. In the gyrokinetic approach [13], the dynamics of the high frequency cyclotron motion of the charged particles in the strong magnetic field is averaged out, reducing the six-dimensional equation to a five-dimensional "gyrokinetic" equation:

$$\frac{df_\alpha}{dt} = \frac{\partial f_\alpha}{\partial t} + \frac{d\mathbf{R}}{dt} \cdot \frac{\partial f_\alpha}{\partial \mathbf{R}} + \frac{dv_\parallel}{dt}\frac{\partial f_\alpha}{\partial v_\parallel} = 0, \qquad (1)$$

where $f_\alpha(\mathbf{R}, v_\parallel, \mu)$ is the five-dimensional phase space distribution function for species α in the gyrocenter coordinates $\mathbf{R}$, $v_\parallel$ is the velocity parallel to the magnetic field, and μ is the magnetic moment. In the electrostatic case, this equation together with Poisson's equation describes the dynamics of a plasma of our interest.

GTC [5, 6, 17] is an efficient and highly-parallel PIC code that solves the five-dimensional gyrokinetic equation in full, global torus geometry (Figure 1) of typical tokamak fusion devices. The equilibrium magnetic geometry is described by a large aspect ratio analytical model of a simplified toroidal magnetic field with a circular cross-section. This model takes into account all the toroidicity effects, such as the curvature drift and multiple rational surfaces, but not the non-circular cross-section effects or the fully electromagnetic, non-adiabatic electron dynamics found in some of the other gyrokinetic codes [2, 12, 25]. Such an approach enables us to work with a sufficiently simple but nevertheless complete physics model for studying the scaling of turbulent transport spanning the range from present generation experiments to the large ITER-scale plasmas [15, 22]. Specifically, this approach includes all of the important physics captured in numerous global PIC simulation studies of plasma size scaling over the years extending from the work by Z. Lin, et al. [15], up to the more recent work by B. F. McMillan, et al. [22] on system size effects on turbulent transport. These techniques reduce the complexity of developing the algorithmic advances required to take advantage of rapidly evolving architectural platforms.

GTC utilizes a highly specialized grid that follows the magnetic field lines as they twist around the torus. This allows the code to retain the same accuracy while using fewer

toroidal planes than a regular, non field-aligned grid. Further, considering that high frequency wave numbers parallel to the magnetic field are dampened by *Landau damping*, increasing grid resolution in the toroidal dimension leaves the results essentially unchanged. Thus, a production simulation generally consists of only 64 poloidal planes (*ntoroidal* = 64) wrapped around the torus. Each poloidal plane is represented by an unstructured grid with uniform spacing in the radial dimension (*psi*) and uniform spacing in the poloidal dimension (*theta*) for fixed *psi*.

In the gyrokinetic approach, the helical motion of a charged particle in a strong magnetic field is approximated by a charged ring. Thus, in the gyrokinetic PIC method, a particle is not a point object but rather a discrete 4-point representation of a ring of variable (gyro)radius [13]. In the charge deposition and field interpolation phases, the charge and field at a given location are interpolated to/from the 8 nearest grid points (Figure 2). Since each ring spans up to 16 radii, the gyrokinetic PIC method can place immense pressure on the memory subsystem due to poor spatial and temporal locality in the requisite gather/scatter operations.

2.1 Computational Kernels

GTC essentially involves six computational kernels:

Charge deposits charge from particles onto the grid using the 4-point gyro-averaging method. The charge phase of GTC's PIC primarily operates on particle data, but involves the complex particle-grid interpolation step. Particles, represented by a 4-point approximation for a charged ring, deposit charge onto as many as 32 unique grid memory locations. In a shared memory environment, these increments must be either guarded with a synchronization mechanism to avoid read-after-write data hazards or redirected to private copies of the charge grid. The charge deposition phase will also involve MPI communication in a distributed memory environment. For example, particles deposit charge on the two poloidal grids consisting of the left and the right boundary of a toroidal section. If domain is decomposed in toroidal dimension, the charge from neighboring toroidal sections needs to be merged. Additionally, the charge phase includes a `MPI_Allreduce` to obtain a flux-surface-averaged charge for solving the so-called "zonal flow".

Poisson/Field/Smooth solves the gyrokinetic Poisson equation, computes an electric field, and smooths the charge and potential with a filter on the grid. These three steps constitute purely grid-related work in which the floating-point operations and memory references scale with the number of poloidal grid points (*mgrid*). As in the charge deposition phase, in a distributed memory environment, **field** and **smooth** involve MPI communication to update the values in the ghost zones in the toroidal direction (*zeta*). Note that in the gyrokinetic ordering, we usually neglect the three dimensional Debye shielding term in the gyrokinetic Poisson's equation as the Debye shielding term is much smaller than the ion polarization term. Thus, we only need to solve a two-dimensional Poisson's equation in each poloidal plane.

Push interpolates the electric field onto particles and using that field advances particle phase space positions. In the push phase, the electric field values at the location of the particles is "gathered" and used for time-advancing their phase space coordinates. This step also requires reading irregular grid locations in memory (the electric field values) corresponding to the four bounding boxes of the four points on the ring, involving data reads from up to 32 unique memory locations. Fortunately, the operations in the push step are usually independent (hazard free) and are thus relatively less challenging to parallelize.

Shift moves particles between processes in distributed memory environment. In general, particle-related work such as **charge**, **push**, and **shift** generally scales linearly with the number of particles. Grid-related work such as **poisson, smooth** and **field** scales with the number of poloidal grid points (*mgrid*). Simulations typically employ high particle densities, ensuring the time spent in grid-related operations is a minority component of overall runtime.

2.2 Parallelization

Machines such as the 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. Clearly, decomposition solely among the 64 poloidal planes in the toroidal dimension is insufficient. To address this, GTC included three levels of parallelism. First, a one-dimensional domain decomposition occurs in the toroidal dimension *zeta*. As mentioned above, this partitioning is limited by Landau damping effects. In order to further increase MPI parallelism a second level of decomposition, over the particles only, is introduced. Within each toroidal domain, the particles are divided between several processes wherein each process owns a fraction of the total particles in that domain as well as a private copy of the local toroidal grid (usually two poloidal planes) to simplify the charge deposition. To merge these private copies together, GTC performs an `MPI_Allreduce`. The third level of parallelism in the original code is an intra-node shared memory partitioning (via OpenMP) of both particle and grid-related work.

The above implementation shows near-perfect scaling with the number of particles, but not with the number of grid points. This is not an issue for small size plasmas. However, for large fusion devices, such as ITER, which is 8 times larger in volume than the largest fusion device currently in existence, this results in a lower particle density in a simulation assuming the same hardware resource, i.e., memory. The time spent on the grid-related phases dramatically increases and can even dominate the overall running time. In addition, grid sizes are so large that they fall out of cache and degrade performance due to increased cache misses.

Recently, a key additional level of domain decomposition in the radial dimension was introduced to GTC code [1, 4] (GTC-P), and shown to be essential for efficiently carrying out simulations on large scale plasmas. The radial domain decomposition begins by partitioning a poloidal plane to non-overlapping domains with equal area. Assuming particle density is uniform, this partitioning divides all particles in one toroidal section equally across multiple processes. Next, the non-overlapping domain is extended to line up with the mesh boundary in the radial direction (shown as valid grid in Figure 2). Finally, the valid grid is extended on each side with ghost cells accounting for charge deposition with 4-point approximation (shown as local grid in Figure 2). In general, 3 to 8 ghost cells are sufficient. The 2D domain decomposition is implemented with MPI using two different communicators: a toroidal communicator and a radial communicator. The particles move between domains with nearest-neighbor communication. Since the number of particles crossing the radial domain boundaries is much smaller

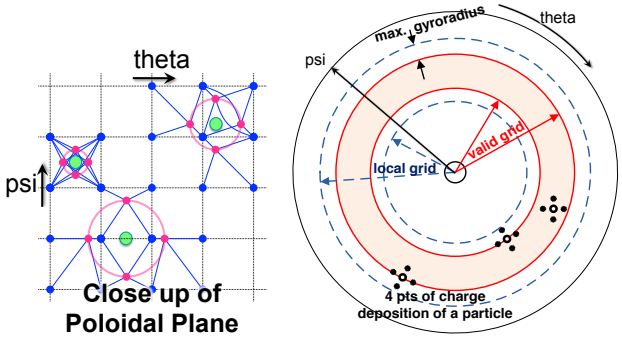

Figure 2: The "4-point gyrokinetic averaging" scheme employed in the charge deposition and push steps (left). A geometric radial partitioning with extended "ghost" zones (right). Valid region contains particle's guiding center for a processor assignment. "Local" extends valid region based on the maximum gyroradius of updates.

Core Architecture	IBM A2	Intel SNBe	AMD Opteron
Clock (GHz)	1.6	2.60	2.1
DP GFlop/s	12.8	20.80	8.4
Data Cache (KB)	16	32+256	64+512
Memory Parallelism	HW prefetch	HW prefetch	HW prefetch
Processor Architecture	**IBM BGQ**	**Cray XC30**	**Cray XE6**
Cores per chip	16	8	6
Last-level Cache	32 MB	20 MB	5 MB
DP GFlop/s	204.8	166.4	50.4
STREAM Bandwidth	28 GB/s	38 GB/s	12 GB/s
Memory Capacity	16 GB	32 GB	8 GB
System	**Mira**	**Edison**	**Hopper**
CPUs/Node	1	2	4
CPUs/NIC	1 (SOC)	8	8
Nodes	49,152	664	6,384
Interconnect	Custom	Aries	Gemini
Topology	5D Torus	Dragonfly	3D Torus

Table 1: Overview of Evaluated Platforms.

than the number of particles crossing the toroidal domain boundaries, radial partitioning results in minimal additional communication.

With a 2D domain decomposition and particle decomposition, GTC-P pushes the scalability of the PIC method to an extreme and can easily scale to the largest systems currently available. The optimal balance between processes in the radial dimension and processes used for particle decomposition is dependent on both the machine and the underlying physics. For example, in a delta-f method with 100-500 particles per cell, particle decomposition is usually not necessary. However, in a full-f method with 5000-8000 particles per cell, all four levels of parallelism must be employed. Conversely, radial decomposition increases network bandwidth requirements but reduces memory usage and reduces cache and DRAM working sets.

3. EVALUATED PLATFORMS

In this paper, we examine performance on three supercomputing systems: the IBM Blue Gene/Q (Mira) at the Argonne Leadership Computing Facility (ALCF), as well as both the Cray XE6 (Hopper) and the Cray XC30 (Edison) at NERSC. In all cases, we use a hybrid MPI+OpenMP programming model that exploits hardware capabilities for fine-grained communication and avoids excessive replication of data.

Mira: is an IBM Blue Gene/Q system at ALCF. The Blue Gene/Q architecture boasts extreme energy efficiency and scalability without sacrificing conventional (MPI or MPI-OpenMP) programming models. In order to maximize energy efficiency, IBM uses an embedded core augmented with a 256-bit SIMD unit underclocked to 1.6GHz. Each core is dual-issue, 4-way multithreaded, and has a 16KB L1 data cache. Multithreading is both a boon and bane for programming and performance. In order to dual-issue, at least two threads must be running per core. Additional threads can be used to hide some of the L1 and floating-point latency. However, threads will contend for the finite L1 capacity and generate contention in the L1 prefetchers. Each BG/Q compute chip contains 17 cores (16 available to the user) connected via a crossbar to a 32MB L2 and the NIC. Although this large cache is helpful in capturing a working set on chip, its latency (in excess of 80 cycles) demands codes attain high L1 locality and efficient use of the L1 prefetchers. The two memory controllers per chip provide a sustained bandwidth of up to 28 GB/s to 16GB of DRAM. There are 1024 nodes per rack and nodes are connected in a high-bandwidth 5D torus. In this paper, we examine GTC-P (new) performance on both Argonne's Mira (49,152 nodes) and Livermore's Sequoia (98,304 nodes).

Hopper: is a Cray XE6 system at NERSC. Unlike the BG/Q architecture, the XE6 is based on commodity AMD processors connected via HyperTransport to custom interconnect. Each processor includes six, 2.1GHz AMD Opteron cores. Each core is superscalar out-of-order, includes 128-bit SIMD (SSE3) floating-point datapaths, and both a 64KB L1 and 512KB L2 cache. Cores are connected to a 6MB L3 cache and two DDR3-800 memory controllers. There are four chips per node and two nodes directly connect to each Cray Gemini NIC. Overall, nodes form a 3D Torus.

Edison: is a Cray XC30 (Cascade) supercomputer at NERSC. The XC30 is based on commodity Intel processors connected via PCIe to custom interconnect. In Phase I of this machine, each processor was an 8-core, 2.6GHz Intel Xeon E5 2670 (SandyBridge-E). The core is superscalar out-of-order, and includes 256-bit SIMD (AVX) floating-point datapaths, as well as a 32KB L1 and 256KB L2 caches. Although each core is dual-threaded, low-latency operations coupled with out-of-order execution often result in little benefit from running two threads per core. Cores are connected via a ring to a 20MB L3 cache and four DDR3-1600 memory controllers. The resultant STREAM bandwidth per socket is more than 33% greater than the Blue Gene/Q platform. There are two chips per node and four nodes directly connect to each NIC in Cray's new Aries Dragonfly interconnect [7].

4. OPTIMIZATION

The original version of GTC-P was developed in FORTRAN 90 using the MPI/OpenMP hybrid programming model. This code demonstrated high scalability to large-size plasma simulations, without excessive memory footprint requirements. Unfortunately, its performance is impeded by load imbalance at high concurrencies. Moreover, since the Poisson solver was implemented via the Portable, Extensible Toolkit for Scientific Computation (PETSc) version 2.3.3, it lacks multithreading capability.

Previous work suggested having a large number of threads per process can enhance the performance of the particle-based components in PIC [19]. As architectures continue to evolve towards manycore, even the grid-based routines can become serial bottlenecks (particularly for the delta-f method where particle density is relatively low). Our work leverages the highly optimized GTC code [19], which addresses the challenges of modern HPC systems built from multi- and many-core processors, to attain node-level scaling. To improve multi-node parallel scalability, we implement the key additional level of radial domain decomposition with special attention to the load imbalance issue. Furthermore, to avoid any serial bottlenecks on multi- and many-core architectures, PETSc 2.3.3 was replaced with a hand-coded multithreaded Poisson solver.[1] One of the motivations in writing our own solver instead of relying on any third party libraries is for portability across the wide variety of supercomputer installations across the world. Finally, we introduce a novel two-level particle binning algorithm for charge deposition to reduce temporary memory usage in multithreaded environments.

The optimized GTC code involved a complete rewrite of the entire GTC Fortran code into C. There are several reasons of using C. First, it simplifies the porting of the code to GPU, Intel Xeon Phi, and upcoming multicore technologies. Second, the rewrite enabled easier exploitation of low-level optimizations. Inherited from the optimized GTC code, the new GTC-P code is also written in C. It is worth emphasizing that the optimized GTC and the newly developed GTC-P codes have the same physics capability as the corresponding Fortran versions. In theory, all optimizations developed in the C version of the codes could be applied to the Fortran versions.

Multicore Optimizations: Data layout and memory access pattern are paramount to performance. GTC principally works on two data representations, particle data and grid data. The original GTC-P code uses arrays of structures for particle data, where each particle element includes 12 double-precision entries (time integrator is the second order Runge Kutta method). As in the previous work [19–21], a structure of arrays (SoA) data layout is chosen for particle data to maximize spatial locality for streaming accesses. Moreover, since the structure of arrays data layout is friendly for SIMD architectures such as BG/Q, GPUs, Opteron, Xeon, Xeon Phi and vector computers, selection of a structure of arrays data layout can maximize performance on a wide range of processor architectures.

The previous single-node work [20, 21] explored optimizing GTC's charge deposition via static replication of grid segments grid, coupled with synchronization via atomics, where the size of the replica may be traded for increased performance. For smaller multicore architectures, the best performance was often obtained by employing the full poloidal grid replication for each OpenMP thread in the hybrid configurations (MPI/OpenMP). However, on BG/Q with 64 OpenMP threads, the full poloidal grid replication strategy dramatically increases the temporary grid-related storage for large size grid (e.g., 2GB for ITER size simulation). The radial domain decomposition solves locality and memory pressure without resorting to costly atomics. In essence, as only a small portion of the full poloidal grid is required for a hazard-free charge deposition, we may employ the private grid replication strategy on a per thread basis for charge deposition.[2]. The charge densities on each private copy are summed at the end of the charge phase, where the order of summation is chosen to minimize synchronization.

Other optimizations include loop-fusion to improve computational intensity, fusing OpenMP loops wherever possible to minimize parallel thread fork-join overhead, flattening 2D and 3D grid arrays to 1D, zero-indexed arrays in C, and pre-allocation of memory buffers that are utilized for temporary storage in every time step.

Optimizations for grid-based phases: The original GTC-P code showed load imbalance issue for grid-based work, e.g., **smooth**, **field** and **poisson**, at high concurrencies under both distributed memory MPI and shared memory OpenMP. The MPI load imbalance is a result of domains belonging to processes which include large ghost zones in *psi* to account for the potentially-large radius of the gyrating particle. In a circular geometry, the domains close to the edge of the plasma have a much larger number of grid points for the same number of ghost cells as those close to the core of the plasma. In GTC-P, the grid-based operations are applied to all local grid points including the ghost zones. The values at the ghost cells are then updated through communication. Observing that the values on the ghost cells will ultimately be updated through communication, we restrict the computation in the grid-based routines to only the grid points of non-overlapping regions (non-ghost zones). This straightforward strategy dramatically reduces the load imbalance issue in the computation stage.

The grid-based subroutines usually include irregularly nested loops in which *psi* is the outer loop and *theta* is the inner loop. On the outermost domains, there is little variability in the loop bounds for *theta*. However, there is a huge variability for *theta* on the innermost domains. Simply threading over *psi* will lead to significant load imbalance. Unfortunately, the `omp collapse(2)` is not applicable for this loop structure. As such, the nested loop was manually flattened into a single loop over all grid points. Any temporary data nominally calculated in the outer loop are now pre-computed and stored in an auxiliary array.

A two-level binning algorithm: It is challenging to achieve highly efficient parallelism for a PIC code due to random access and potential fine-grain data hazard that serializes parts of the computation. Particle binning is usually adopted to improve locality, and data hazards are avoided by providing a private copy of the grid for each thread. However, in the gyrokinetic PIC method, even with particle binning, locality is not guaranteed since each particle represents a charge ring of variable gyroradius. We introduce a two-level binning algorithm to further improve locality and reduce memory footprint for gyrokinetic PIC method. The first level of binning is for particles based on their position.

[1]The hand-coded solver is a damped Jacobi iterative solver in which the damping parameter was chosen to favor the desired range of wavelengths for the fastest growing modes in the simulation of plasma turbulence [18]. The number of iterations was fixed at 5 for all problem sizes as convergence stops due to low frequency error. Numerical observations suggest that the use of a preconditioner like GMRES does not affect the physics results.

[2]By using private grid replication strategy, each thread will carry a private copy of the local grid, thus, avoid potential memory conflict.

Configuration	Torus Shape	Grouping
256 nodes (1/4 rack)	4 2 4 4 2	ABCE×D
512 nodes (1/2 rack)	4 4 4 4 2	ABC×DE (default)
1024 nodes (1 rack)	4 4 4 8 2	ABC×DE (default)
2048 nodes (2 racks)	4 4 4 16 2	ABC×DE (default)
4096 nodes (4 racks)	4 4 8 16 2	ACE×BD
8192 nodes (8 racks)	4 4 16 16 2	AC×BDE

Table 2: Process Mapping on BG/Q

The second level of binning is for updates based on the positions of the affected cells.

a) *Particle-binning:* We leverage the multithreaded radial binning routine from the previous work [19], where the radial indices of particles are used as the sort key. Since we can use the auxiliary array that stores particle data from previous step, the binning can be implemented out-place without additional cost in terms of memory. We also extend the algorithm to bin particles in both the poloidal and radial dimensions. Instead of using the radial indices, we use the one-dimensional grid point indices in a poloidal plane as the sort key. Overall, the best performance is obtained when only binning in the radial dimension. The frequency of radial binning is a tuning parameter that can be set dynamically. For all our experiments in §5, particles are only binned in the radial dimension and the binning frequency is set to one (we apply binning at every time step).

b) *Update-binning:* Besides binning gyroparticles according to their coordinates periodically, we also bin the four points of all gyroparticles according to their coordinates at every time step. Specifically, charge deposition and field interpolation are completed in two stages. In the first stage, we compute the weights and the coordinates of four points, re-order and store them in a new array. In the second stage, we stream the 4-point array, apply charge deposition to their 8 nearest grid points. With point binning, a local grid private to each thread spans up to 1 radii instead of 16. Due to reduced ghost zones, the point binning approach significantly reduces the temporary memory storage for hazard-free charge deposition. Compared with the algorithm that uses auxiliary arrays to store neighboring grid points for avoiding redundant calculation in **push**, storing 4-point weights and coordinates does not increase memory requirement.

Note that due to redundantly calculating the neighboring grid points in **push**, the point binning algorithm does not lead to an overall performance improvement for the problem configurations and computer systems in this paper. However, this will be an important optimization for architectures featuring massive on-chip parallelism and low memory per core, such as GPUs, where redundant computations in **push** can easily be hidden by massive chip-level parallelism.

Blue Gene/Q optimizations: GTC-P uses a two-dimensional topology for point to point communication, where the first dimension (toroidal dimension) has fixed dimensionality 64. On a BG/Q system with 5D torus network, we can thus group two or three torus dimensions together to match 64 for an optimized placement layout by setting the environment variable `RUNJOB_MAPPING`. Table 2 shows some examples of configurations and groupings when we run the application with one process per node. It should be noted that offline experiments showed this explicit process mapping improved communication time by 45% for particle shift in the toroidal dimension when using 8 racks (8192 nodes) of Mira.

Grid Size	A	B	C	D
mpsi	90	180	360	720
mthetamax	640	1280	2560	5120
mgrid (grid points per plane)	32449	128893	513785	2051567
chargei grid (MB)†	0.5	1.97	7.84	31.30
evector grid (MB)†	1.49	5.90	23.52	93.91
Total particles *micell*=100 (GB)	0.29	1.16	4.64	18.56
Total particles *micell*=500 (GB)	1.45	5.8	23.2	92.8

Table 3: The GTC numerical settings for different plasma sizes. The grid and particle memory requirements are for one toroidal domain only. A simulation typically consists of 64 toroidal domains.

As BG/Q is a highly multithreaded architecture (up to 64 OpenMP threads per process), efficient use of OpenMP is essential for attaining high performance. Although significant performance gains can be made in the code, like MPI, environment variables can significantly boost performance for some routines. The variables `BG_SMP_FAST_WAKEUP=YES` and `OMP_WAIT_POLICY=active` make threads use an atomic-based spin barrier instead of a slower sleep-based approach. For some of the grid-based routines, these settings resulted in a 10× speedup when using 64 threads.

5. SIMULATION RESULTS AND ANALYSIS

In this section, we present the performance for both strong and weak scaling experiments on different plasma sizes using 100 or 500 particles per cell (labeled *micell*). Using a delta-f method, *micell*=100 is a typical particle density number used for production runs. The purpose of exploring the performance for *micell=500* is twofold. First, we use higher particle density numbers for a convergence study. Second, exploring higher particle density numbers will help us understand the performance of the code with a full-f method in which a 10× increase in particle number density is required to reduce numerical noise.

5.1 Configuration

A GTC simulation is described by several important numerical parameters: the toroidal grid size *ntoroidal*, the poloidal grid size *mgrid*, the average particle density as measured in the ratio of particles to grid points *micell*. Thus, the total number of particles in a simulation is $ntoroidal \times mgrid \times micell$. In order to demonstrate the viability of our optimizations across a wide variety of potential simulations, we explore four different grid problem sizes, labeled *A, B, C, D*, and two different resolutions (particle densities): 100 and 500 particles per cell. Grid size *A* and *B* correspond to the majority of existing tokamaks in the world, *C* corresponds to the JET tokamak, the largest device currently in operation [11], and *D* to ITER. Table 3 lists the poloidal grid sizes and the corresponding grid and particle related memory requirement in one toroidal domain for different plasma sizes. Observe that when moving to a plasma device of one size larger, e.g., *A* to *B*, the poloidal grid size and number of particles in one toroidal domain increase by 4×.

We perform three series of scaling studies to demonstrate the parallel efficiency of the code on leadership-class systems, with a focus on BG/Q system (Mira, Sequoia). To demonstrate the flexibility of the code as well as cross-architectural comparisons, we provide performance and scaling

MPI Ranks	# of Radial Partitions	GTC-P (original)	Eff.	GTC-P (new)	Eff.	Speed Up
2048	32	9.282	1.0	5.275	1.0	1.76×
4096	64	4.685	0.99	2.651	0.99	1.76×
8192	128	2.536	0.92	1.373	0.96	1.85×
16384	256	1.453	0.80	0.726	0.91	2.00×
32768	512	0.873	0.60	0.414	0.80	2.21×

Table 4: Wall-clock time (sec) for one time step with strong scaling in radial domain decomposition for D100 using GTC-P (original) and GTC-P (new). In all experiments, we use 4 processes/node and 16 threads/process. The new code attains a 2× speedup due to our optimizations.

results on two additional platforms, the Cray XE6 system (Hopper) and the Cray XC30 (Edison).

5.2 Strong Scaling

In the first study, we perform strong scaling experiments with the radial domain decomposition for the D100 problem, an ITER size simulation with resolution of 100 particles per cell. This problem consists of 64 toroidal domains (*ntoroidal*=64) where each domain carries two poloidal planes of 2 million grid points each. Thus, each toroidal domain contains roughly 200 million particles. In all, this corresponds to 130 million total grid points and 13 billion total particles. Our experiments scale the number of radial partitions for each toroidal domain from 32 to 512. On the BG/Q system with 4 MPI processes per node and 16 OpenMP per process, these simulations scale from 512 to 8,192 nodes (8,192 to 131,072 cores).

The scaling of radial domain decomposition for D100 is shown in Table 4. The new version of GTC-P obtains 80% parallel efficiency on up to 32,768 processes. Compared with the original version, GTC-P (new) attains 2.12× overall speedup on 32,768 processes, where the parallel efficiency has increased from 60% to 80% due to our MPI load balance optimization (§4).

The scaling in the radial dimension is limited by the number of grid points in the radial dimension, e.g, *mpsi*=720 for the D problem. If that limit is reached, we can still maintain scalability to an even larger number of cores by enabling the particle decomposition. As discussed earlier in §2.2, the particles within each subdomain can be distributed over multiple processes. In our current version, each process carries a full copy of the partial grid (a subset of the full poloidal grid) and the grid-based work is redundantly computed on these processes. An efficient implementation is now under development which aims at scaling the grid-based work between these multiple processes. Note that we do not use particle decomposition in all the strong scaling experiments in §5.2.

As multi- and manycore processors evolve towards ever larger numbers of cores with small increases in total memory, it is increasingly critical to study the strong scaling properties of the code with respect to thread concurrency. This second series of tests aims at examining the parallel efficiency in terms of the number of threads in a single process, i.e., we fix the problem size and the number of MPI processes, but vary the number of OpenMP threads per process. Again problem D is evaluated, but with two different particle densities — 100 and 500. In order to scale the problem to a large number of cores, a 512-way radial partitioning approach is selected. With 64-way toroidal partitioning, the simulation uses 32,768 processes. Our experiments strong-scale from 2048 to 32,768 BG/Q nodes by increasing the number of OpenMP threads in a single process from 4 to 64. As concurrency increases, we expect the computational time to decrease proportionally, while the workload in a single MPI process remains constant.

Nodes	2048	4096	8192	16384	32768
Thread/Process	4	8	16	32	64
charge	1.2×	2.1×	2.0×	3.5×	8.8×
push	1.7×	1.7×	1.7×	1.7×	1.7×
shift	1.4×	2.1×	2.2×	2.2×	2.9×
poisson	6.8×	12.1×	13.6×	13.6×	16.5×
field	41.0×	35.7×	28.4×	20.3×	20.1×
smooth	28.9×	23.9×	21.7×	16.5×	16.4×
overall speedup	1.4×	1.8×	2.1×	2.9×	5.6×

Table 6: Speedup of GTC-P (new vs. original) on D100 problem by kernels with different threads per process. The total processes (32,768), the number of particles in each process (400,556), and the number of radial partitions (512) are fixed. Shift includes shift_t, shift_r and binning in Table 5.

Table 5 details wall-clock times on each kernel for one iteration step for the D100 and D500 problems respectively. The results show that **push** and **binning** scale nearly ideally up to 64 threads, an expected result given the parallel nature of these computations. These results also suggests that **push** and **binning** will be strong candidates for emerging manycore systems such as GPU and Xeon Phi. **charge** and **shift** (including the shift in the toroidal dimension **shift_t**, the shift in the radial dimension **shift_r** and **binning**) also show efficient scaling behavior. The slowdown of the scaling at 64 threads is due to the serial communication portion of the code. For grid-based phases, the scaling is saturated at 16 threads, except for **poisson**. Finally, observe that higher particle density leads to improved parallel efficiency.

The speedup of GTC-P (new) vs. GTC-P (original) by kernels with different threads per process is shown in Table 6. Note the significant increase in performance particularly for the grid-based phases due to our optimizations. The performance boost for **charge** and **push** are mainly from improving locality through binning and avoiding costly synchronization in **charge**. Both the original and the new codes employ the private grid replication strategy on a per thread basis for charge deposition. However, GTC-P (original) uses a `critical section` to merge charges from all copies private of threads. This serial portion of the code can be easily avoided by carefully reorganizing the summation order. For **charge** and **shift**, the improved speedup with increased threading suggests that the original code is not well designed for fine-grained parallelism. For example, most memory copies in **shift** are not multithreaded. Overall, the new code delivers up to an impressive 5.6× speedup over the original version.

5.3 Weak Scaling

The parallelism in GTC is denoted as three parameters: the number of toroidal partitions *ntoroidal*, the number of radial partitions *nradiald* and the number of particle de-

D100: 13 Billion particles, 32,768 total processes, with 400,556 particles per process											
Nodes	**Processes×Threads**	**Charge**	**Push**	**Shift_t**	**Shift_r**	**Binning**	**Poisson**	**Field**	**Smooth**	**Total**	**Eff.**
2048	16×4	0.6691	0.4569	0.3073	0.0326	0.0602	0.0197	0.0050	0.0069	1.558	1.0
4096	8×8	0.3397	0.2313	0.1536	0.0169	0.0340	0.0078	0.0027	0.0047	0.791	0.99
8192	4×16	0.1822	0.1178	0.0779	0.0094	0.0167	0.0044	0.0021	0.0039	0.414	0.94
16384	2×32	0.1110	0.0591	0.0722	0.0063	0.0087	0.0033	0.0022	0.0039	0.267	0.73
32768	1×64	0.0709	0.0298	0.0487	0.0050	0.0039	0.0022	0.0018	0.0035	0.166	0.59

D500: 65 Billion particles, 32,768 total processes, with 2,002,780 particles per process											
Nodes	**Processes×Threads**	**Charge**	**Push**	**Shift_t**	**Shift_r**	**Binning**	**Poisson**	**Field**	**Smooth**	**Total**	**Eff.**
2048	16×4									OOM	-
4096	8×8	1.5637	1.1545	0.4108	0.0842	0.1680	0.0077	0.0027	0.0048	3.396	1.0
8192	4×16	0.7972	0.5871	0.1919	0.0445	0.0824	0.0044	0.0020	0.0039	1.713	0.99
16384	2×32	0.3958	0.2943	0.1361	0.0250	0.0423	0.0033	0.0022	0.0039	0.903	0.94
32768	1×64	0.2148	0.1478	0.0082	0.0155	0.0213	0.0022	0.0018	0.0035	0.489	0.87

Table 5: Strong scaling of threads per process for D100 (top) and D500 (bottom) using GTC-P (new) on Mira. The time is the wall-clock time (sec) for one time step. We use 64-way toroidal and 512-way radial partitioning. The D500 problem could not fit in 2048 BG/Q nodes due to memory constraints. OOM stands for Out-of-Memory.

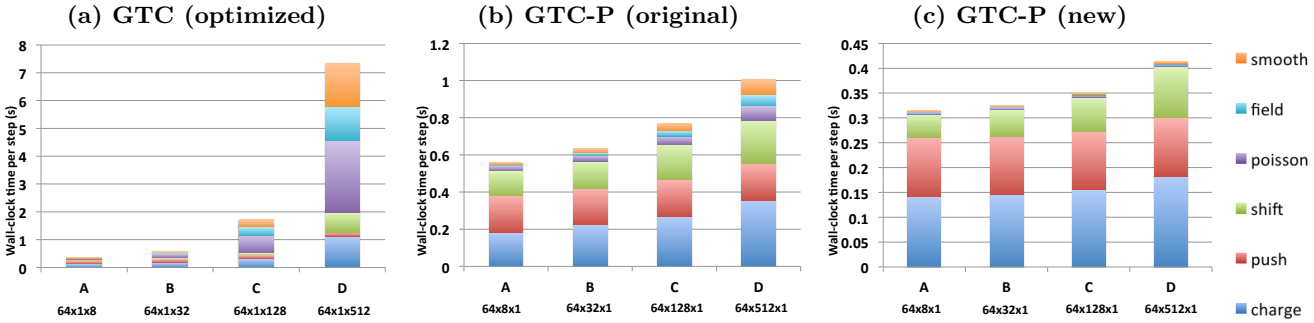

Figure 3: Weak scaling study from A to D plasma size on Mira BG/Q. Colors represent runtime spent in each phase. In all experiments, the number of particles per process is about 404,487. The parallel decomposition for each case is denoted as *ntorodial* × *nradiald* × *npe_radiald*. All experiments use 4 processes/node and 16 threads/process. Note that each plot is on a different timescale, where for case D, GTC-P (new) is 17.68× and 2.11× faster than GTC (optimized) and GTC-P (original) respectively.

composition in one domain *npe_radiald=npartdom/nradiald*, where *npartdom* is the total number of particle partitions in one toroidal section.

GTC employs a 1D toroidal domain decomposition and a particle decomposition. As mentioned earlier, GTC-P is based on GTC but introduces an additional level of decomposition in the radial dimension. Thus it has three approaches of MPI parallelism: a 2D domain decomposition and a particle decomposition.

Figure 3 shows the weak scaling results for A, B, C and D problem sizes with a particle density of 100 using GTC (optimized) [19], GTC-P (original) and GTC-P (new). We start from A size problem with *npartdom*=8. The number of particles in each process is approximately the same. Parameters *mgrid* and *npartdom* are increased proportionally with the number of cores. For GTC (optimized), with $nradiald = 1$, the degree of particle decomposition *npe_radiald* varies from 8 to 512 for problems for sizes A to D. Similarly, for GTC-P (original) and GTC-P (new), with $npe\_radiald = 1$, the number of radial partitions *nradiald* varies from 8 to 512. At size D, where $ntoroidal = 64$, $npartdom = 512$, the total number of processes is 32,768. In contrast to previous weak scaling studies where the grid size were kept constant, our experiments represent a true weak scaling study of both the grid and the particles.

Figure 3 (a) presents weak scaling results for the GTC (optimized) code. Without radial domain decomposition, we see that the runtime for the grid-related phases increases dramatically as the problem size is increased, and ultimately dominates the total computing time. Due to large cache working set and the overhead of merging duplicate copies of the charge grid from all processes in a toroidal section, the charge phase shows large performance degradation when scaled to size D. Figure 3 (b) and (c) present weak scaling results for GTC-P (original) and GTC-P (new), respectively. Compared with the original code, our optimizations in the new version effectively address the load imbalance challenges in the grid-related phases.

In order to explore the scaling to a large number of processor cores, we evaluate the same weak scaling study but with 1 MPI process per node and 64 threads per process on Mira. Figure 4 demonstrates an impressive scaling up to 524,288 cores (32,768 nodes) using GTC-P (new) for size D.

To further explore performance at extreme scale, we conducted a weak scaling study using GTC-P (new) on the BG/Q Sequoia system at LLNL, with the full 96 racks (1.5 million cores). In order to scale the problem to the full Sequoia system, we use 1 process per node and 64 threads per process and turn on the additional level of parallelism via particle decomposition. Particles in each domain are evenly

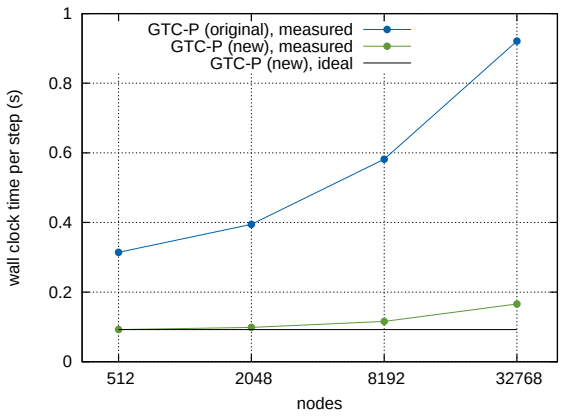

Figure 4: Weak scaling from A to D size plasmas using GTC-P (original) and GTC-P (new) on Mira. This is the same weak scaling as in Figure 3 (b) and (c), but with 1 MPI process/node and 64 threads/process.

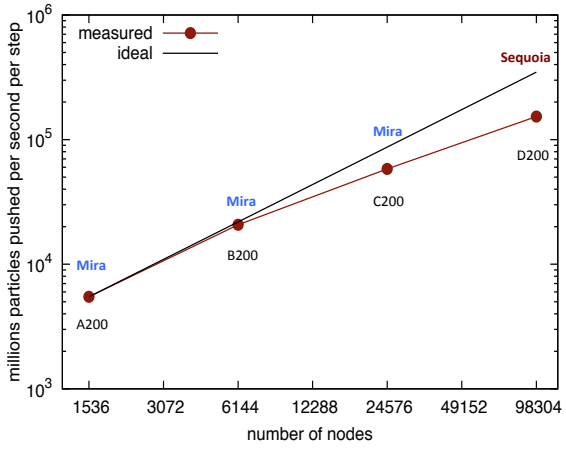

Figure 5: GTC-P (new) weak scaling with respect to "the number of particles pushed per second per step" with different size devices on BG/Q. Particle numbers in one process is about 404,487. *ntorodial*=64, *npe_radiald*=2, *nradiald*=8, 32, 128, and 512 for A,B,C and D respectively. The D size was performed on Sequoia, while the others were on Mira. Experiments use 1 process/node and 64 threads/process.

divided between 2 processes, *npe_radiald*=2. As in Figure 3 and 4, the number of particles in a single process is about 404,487. In Figure 5, we see that GTC-P (new) scales well to 1,572,864 cores on BG/Q.

To demonstrate the flexibility and benefits of our optimized implementation, we evaluated the new GTC-P on two additional large-scale systems at NERSC: the Cray XE6 (Hopper) and the Cray XC30 (Edison). We weak scale B, C and D problem sizes with 100 particles per cell. To fit all problems to the pre-production Edison platform, a 664 compute nodes system, we apply 1-way, 4-way, and 16-way radial partitioning to different problem sizes with the 16-way partitioning for D size. Note that no particle decomposition is applied. Suggested by the previous study [19], we select 1 MPI per chip with in-chip parallelism explored with OpenMP. On BG/Q, this corresponds to 1 MPI process per node and 64 OpenMP threads per process (1 MPI/-64 OpenMP). Hopper and Edison have 4 and 2 CPU chips per compute node, respectively. Therefore, we use 4 MPI processes per node and 6 OpenMP threads per process on Hopper, and use 2 MPI processes per node and 8 OpenMP threads per process on Edison.

Figure 6 shows the performance of the new GTC-P code on Mira, Hopper and Edison. The performance result on Mira is quite flat, a demonstration of excellent weak scaling. On Edison and particularly on Hopper, the time spent in **shift** increases as we increase the number of processors. The results indicate that Mira is better balanced between computation and network communication. We also observe that, despite consuming half the power, Mira(BGQ) delivers about the same performance per chip as Edison(SNBe). Conversely, Edison delivers about 3× better performance per chip compared to Hopper(Opteron).

6. PARTICLE CONVERGENCE STUDY

The primary focus of this research is to capture new physics insights into the key question of how turbulent transport and associated confinement characteristics for large laboratory plasmas such as the ITER-scale burning plasmas. This is a critically important practical issue because in present generation tokamak experiments the Bohm-like unfavorable scaling with plasma size is observed, but the much larger (by more than a factor of 3 greater than the largest current experiment) plasma size expected for ITER has been predicted in earlier simulations to enter the favorable Gyro-Bohm scaling regime [15]. GTC, developed to study turbulence transport in fusion plasmas, has a long history of important scientific discoveries through the state-of-art computers [6, 14, 15, 17]. In the past, due to limitations in the computational resources, most simulations were either focused on small size plasmas with moderate particle resolution or large size plasmas with low particle resolution. With the advances of computer architectures and the development of the modern GTC-P code, we now have the ability to study long time behavior of turbulence transport on large scale devices (ITER-size) with unprecedented phase-space resolution (500 particles per cells). Compared with the original GTC ITER-size simulation of 7000 time steps [15], our production run on Mira has increased the phase-space resolution by up to 65× and total time steps by 4×.

We conducted particle convergence studies for Ion Temperature Gradient (ITG) driven turbulence for ITER-size plasma using 32 racks of Mira. The physical parameters for the simulations are the Cyclone case [3] which has peak ion temperature gradient at $r = 0.5a$ and local parameters: $R_0/L_T = 6.9$, $R_0/L_n = 2.2$, $q = 1.4$ and $(r/q)(dq/dr) = 0.78$. Here a is the minor radius, R_0 is the major radius, L_T and L_n are the temperature and density gradient scale length, respectively. The safe factor q defines the amount of twist in the magnetic field as a function of radius. A homogeneous Dirichlet boundary condition is enforced for the electrostatic potential at the core and edge boundary, $r = 0.1a$ and $r = 0.9a$. This model uses a circular cross section and assumes electrostatic fluctuations with adiabatic electron. The pressure gradient profile, which is the driving force for

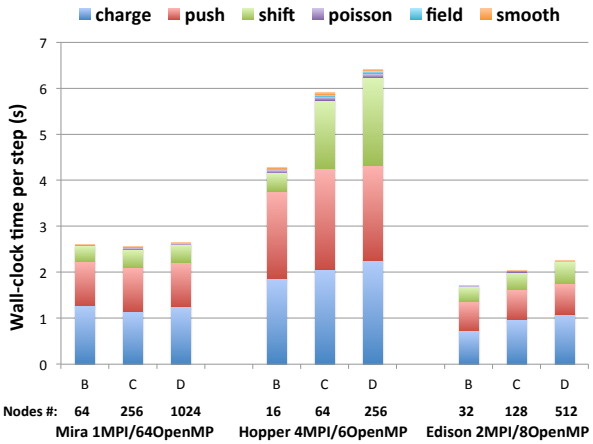

Figure 6: Runtime spent in each phase for B, C and D problems using GTC-P (new) on three different architectures. *ntoroidal*=64, *npe_radiald*=1 and *nradiald*=1,4 and 16 for B, C and D. Particle number in one process is 12,871,300 for all tests.

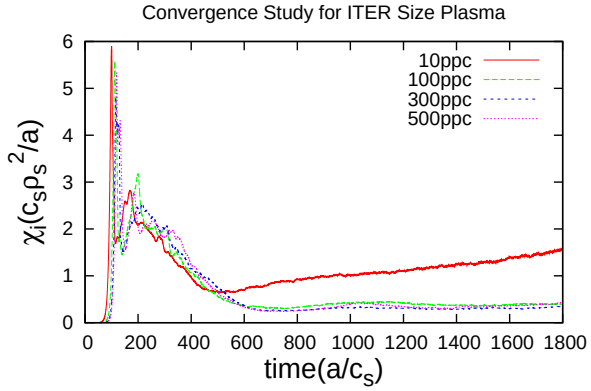

Figure 7: The time evolution of the heat conductivity in Gyro-Bohm units for a wide range of particle resolutions. a is the minor radius of the tokamak and $c_s(=\sqrt{Te/m_i})$ is the ion acoustic speed. The normalized time 1800 corresponds to 30,000 time steps.

the turbulence, is defined as $\exp\{-[r-0.5a]/0.35a]^6\}$. The velocity space nonlinearity, which is usually ignored as a high order term, but has been shown to be important to maintain energy conservation [3], is included in the simulations. For simplicity, the simulations presented here do not include any heat bath or collision model to prevent profile relaxation and stabilize long time simulations. A more in depth study of the effects of various dissipation models and collisions at ITER-scale will be presented elsewhere.

In our experiments, particle number convergence studies using 10, 100, 300 and 500 particles per cell (ppc) were performed for an ITER-size plasma (Figure 7). A total of up to 65 billion particles were evolved for 30,000 time steps. The result shows a convergent thermal flux with particle density greater than 100. This indicates that 10ppc resolution is not sufficient to obtain accurate results for long time simulations, where we see an unphysical enhancement of the thermal flux due to numerical noise. The use of a heat bath or other phase space dissipation models can help alleviate this problem. It is important to highlight that the most expensive case (500 ppc) took only 4 hours using 32 racks of Mira (2/3 of the full system).

The particle number convergence studies are the first essential step toward accurate turbulent transport prediction. With the modern GTC-P capability engaged with BG/Q at scale, we are able to study a variety of important problems systematically with high-fidelity simulations, e.g., the very interesting proposition that the widths of the turbulent eddies generated are insensitive to the size of the plasma. Also, with regard to associated physics properties, theoretical models of "turbulence spreading" [8, 16] have been proposed following the findings in [15]. However, systematic simulation studies have not been able to be carried out to ascertain the true underlying turbulent transport properties of the plasma as the system/plasma size increases from the observed Bohm-like to the predicted future Gyro-Bohm-like regimes. In addition, the more recently proposed explanation that plasma size scaling is determined primarily by the finite width of the strong gradient region [22] can be further assessed via systematic large-scale simulations capable of simultaneously resolving the meso and micro scales.

7. CONCLUSIONS

We present our efforts in developing a new GTC-P code for gyrokinetic plasma simulations. Excellent performance with respect to "time to solution" has been demonstrated on IBM Blue Gene/Q with the full 786,432 cores of Mira at the ALCF and recently with the full 1,572,864 cores of Sequoia at LLNL. Leveraging from previous GPU exploitation of GTC [9, 19], we will implement the GPU capability based on our new GTC-P code. The hybrid code will be deployed and aggressively exercised on LCF-class GPU/CPU hybrid systems such as Titan at the OLCF.

The new GTC-P code, engaged with leadership computing platforms, is critically important in the acquisition of new scientific insights into the physics of turbulent transport in ITER-sized plasmas that can only be achieved by large-scale, high-resolution simulations. In addition, the knowledge and experience presented in this paper will help introduce more comprehensive gyrokinetic PIC codes (including those with more geometric complexity and those that account for electromagnetic dynamics) into the low memory per core path to the exascale era of modern supercomputing applications. It should also be noted that the "scientific discovery" and "transformational" aspect of such studies is that, while the simulation results can be validated against present-day tokamaks, there are no existing devices today that have a minor radius of even one-third the size of ITER. Accordingly, the role of high-fidelity predictive simulations takes on even more importance, especially since the expected improvement in confinement for ITER-sized devices cannot be experimentally validated until after it is constructed and operational.

Acknowledgments

Dr. Wang was supported by the NSF OCI-1128080/G8 Initiative: G8 Research Councils Initiative on Multilateral Re-

search Funding. Authors from Princeton Plasma Physics Laboratory were by the DOE contract DE-AC02-09CH11466. Authors from Lawrence Berkeley National Laboratory were supported by the DOE Office of Advanced Scientific Computing Research under contract number DE-AC02-05CH-11231. Dr. T. Williams was supported by the DOE contract number DE-AC02-06CH11357. An award of computer time was provided by the Innovative and Novel Computational Impact on Theory and Experiment (INCITE) program. This research used resources of the Argonne Leadership Computing Facility, which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357. This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. The authors would also like to thank the NNSA for access to the Sequoia system at LLNL. Sequoia is dedicated to NNSA's Advanced Simulation and Computing (ASC) program for stewardship of the nation's nuclear weapons stockpile, a joint effort by LLNL, Los Alamos National Laboratory and Sandia National Laboratories. ASC advances high performance computing for national security, related science and engineering, and other national challenges.

References

[1] M. F. Adams, S. Ethier, and N. Wichmann. Performance of particle in cell methods on highly concurrent computational architectures. *Journal of Physics: Conference Series*, 78(1):012001, 2007.

[2] J. Candy and R. E. Waltz. An eulerian gyrokinetic-maxwell solver. *J. Comp. Phys.*, 186:545–581, 2003.

[3] A. M. Dimits, G. Bateman, M. A. Beer, B. I. Cohen, W. Dorland, G. W. Hammett, C. Kim, J. E. Kinsey, M. Kotschenreuther, A. H. Kritz, L. L. Lao, J. Mandrekas, W. M. Nevins, S. E. Parker, A. J. Redd, D. E. Shumaker, R. Sydora, and J. Weiland. Comparisons and physics basis of tokamak transport models and turbulence simulations. *Physics of Plasmas*, 7(3):969–983, 2000.

[4] S. Ethier, M. Adams, J. Carter, and L. Oliker. Petascale parallelization of the Gyrokinetic Toroidal Code. In *Proc. High Performance Computing for Computational Science (VECPAR'10)*, 2010.

[5] S. Ethier, W. M. Tang, and Z. Lin. Gyrokinetic particle-in-cell simulations of plasma microturbulence on advanced computing platforms. *Journal of Physics: Conference Series*, 16:1–15, 2005.

[6] S. Ethier, W. M. Tang, R. Walkup, and L. Oliker. Large-scale gyrokinetic particle simulation of microturbulence in magnetically confined fusion plasmas. *IBM Journal of Research and Development*, 52(1-2):105–115, 2008.

[7] G. Faanes, A. Bataineh, D. Roweth, T. Court, E. Froese, B. Alverson, T. Johnson, J. Kopnick, M. Higgins, and J. Reinhard. Cray Cascade: a scalable HPC system based on a Dragonfly network. In *Proc. Int'l. Conf. on High Performance Computing, Networking, Storage and Analysis (SC '12)*, pages 103:1–103:9, Los Alamitos, CA, USA, 2012. IEEE Computer Society Press.

[8] O. D. Gurcan, P. H. Diamond, T. S. Hahm, and Z. Lin. Dynamics of turbulence spreading in magnetically confined plasmas. *Physics of Plasmas*, 12(3):032303, 2005.

[9] K. Z. Ibrahim, K. Madduri, S. Williams, B. Wang, S. Ethier, and L. Oliker. Analysis and optimization of gyrokinetic toroidal simulations on homogenous and heterogenous platforms. *International Journal of High Performance Computing Applications*, 2013.

[10] The ITER project. `http://www.iter.org/`, last accessed Apr 2013.

[11] Joint European Torus (JET). `http://www.efda.org/jet/`, last accessed Apr 2013.

[12] S. Jolliet, A. Bottino, P. Angelino, R. Hatzky, T. Tran, B. Mcmillan, O. Sauter, K. Appert, Y. Idomura, and L. Villard. A global collisionless pic code in magnetic coordinates. *Comput. Phys. Commun.*, 177(5):409, 2007.

[13] W. Lee. Gyrokinetic particle simulation model. *Journal of Computational Physics*, 72(1):243–269, 1987.

[14] W. W. Lee, S. Ethier, R. Kolesnikov, W. X. Wang, and W. M. Tang. Nonlinear turbulent transport in magnetic fusion plasmas. *Computational Science and Discovery*, 1(1):015010, 2008.

[15] Z. Lin, S. Ethier, T. S. Hahm, and W. M. Tang. Size scaling of turbulent transport in magnetically confined plasmas. *Phys. Rev. Lett.*, 88:195004, Apr 2002.

[16] Z. Lin and T. S. Hahm. Turbulence spreading and transport scaling in global gyrokinetic particle simulations. *Physics of Plasmas*, 11(3):1099–1108, 2004.

[17] Z. Lin, T. S. Hahm, W. W. Lee, W. M. Tang, and R. B. White. Turbulent transport reduction by zonal flows: Massively parallel simulations. *Science*, 281(5384):1835–1837, 1998.

[18] Z. Lin and W. W. Lee. Method for solving the gyrokinetic poisson equation in general geometry. *Phys. Rev. E*, 52:5646–5652, Nov 1995.

[19] K. Madduri, K. Z. Ibrahim, S. Williams, E.-J. Im, S. Ethier, J. Shalf, and L. Oliker. Gyrokinetic toroidal simulations on leading multi- and manycore HPC systems. In *Proc. Int'l. Conf. for High Performance Computing, Networking, Storage and Analysis (SC '11)*, pages 23:1–23:12, New York, NY, USA, 2011. ACM.

[20] K. Madduri, E. J. Im, K. Ibrahim, S. Williams, S. Ethier, and L. Oliker. Gyrokinetic particle-in-cell optimization on emerging multi- and manycore platforms. *Parallel Computing*, 37(9):501–520, 2011.

[21] K. Madduri, S. Williams, S. Ethier, L. Oliker, J. Shalf, E. Strohmaier, and K. Yelick. Memory-efficient optimization of gyrokinetic particle-to-grid interpolation for multicore processors. In *Proc. ACM/IEEE Conf. on Supercomputing (SC 2009)*, pages 48:1–48:12, Nov. 2009.

[22] B. F. McMillan, X. Lapillonne, S. Brunner, L. Villard, S. Jolliet, A. Bottino, T. Görler, and F. Jenko. System size effects on gyrokinetic turbulence. *Phys. Rev. Lett.*, 105:155001, Oct 2010.

[23] NuFuSE. `http://www.nu-fuse.com/`.

[24] W. Tang and D. Keyes. Scientific grand challenges: Fusion energy science and the role of computing at the extreme scale. In *PNNL-19404*, page 212, 2009.

[25] W. X. Wang, Z. Lin, W. M. Tang, W. W. Lee, S. Ethier, J. L. V. Lewandowski, G. Rewoldt, T. S. Hahm, and J. Manickam. Gyro-kinetic simulation of global turbulent transport properties in tokamak experiments. *Phys. Plasmas*, 13:092505, 2006.