Benchmark Tests of Fusion Plasma Simulation Codes for Studying Microturbulence and Energetic-Particle Dynamics

Tomo-Hiko WATANABE[1,2)], Yasushi TODO[1,2)] and Wendell HORTON[3)]

[1)]*National Institute for Fusion Science, Toki, Gifu 509-5292, Japan*
[2)]*The Graduate University for Advanced Studies, Toki, Gifu 509-5292, Japan*
[3)]*Institute for Fusion Studies, The University of Texas at Austin, Austin, Texas 78712, USA*

(Received 25 July 2008 / Accepted 3 November 2008)

Benchmark tests of two simulation codes used for studying microturbulence and energetic-particle dynamics in magnetic fusion plasmas are conducted on present-day parallel supercomputer systems. Both the codes achieved high efficiency on the Earth Simulator with vector processors, and showed good performance scaling on massively parallel supercomputers with more than 10,000 commodity processors. The benchmark results obtained indicated high adaptability of fusion plasma simulation codes to state-of-the-art supercomputer systems.

Keywords: computer simulation, fusion plasma, high performance computing, supercomputer

DOI: 10.1585/pfr.3.061

1. Introduction

Computer simulations performed using massively parallel supercomputer systems have revealed new insights and led to discoveries in various fields of science. In magnetic fusion research, large-scale numerical simulations have been regarded as indispensable tools for investigating nonlinear dynamics of magnetically confined plasmas and predicting their complex behaviors [1, 2]. To promote the simulation studies effectively, it is quite important to find theoretical models for describing physical phenomena and to develop efficient algorithms and numerical codes.

Most of the present-day supercomputer systems employ machine architectures with parallel computation nodes and distributed memory. Parallel architecture demands the use of efficient computation algorithms and the implementation of the message-passing interface (MPI) [3]. Moreover, several processor cores are often placed in a single computation node, sharing memory and the internode connection. Several types of processors (vector or scalar), core architectures (single or multi-core), node structures (single or multi-socket), and internode connections (cross bar, torus, or fat-tree) are adopted in recent supercomputers. Because of these complications in the design and structure of supercomputer systems, it is not clear how efficiently the actual applications can be executed. The effective computational performance of applications can depend strongly on the machine architecture, and this may affect the feasibility of simulation studies. Therefore, the performance of the large-scale simulation codes should be discussed, and this is the focal point of the present study.

This paper provides a technical report on the performance of two major fusion plasma simulation codes: the gyrokinetic Vlasov simulation code (`GKV` code) and the energetic-particle and magnetohydrodynamic (MHD) hybrid simulation code (`MEGA` code). They were originally developed and optimized on the Plasma Simulator (SX-7/160M5) at National Institute for Fusion Science (NIFS). Brief summaries of the simulation codes are given in the next section. We have carried out a series of benchmark tests on six supercomputer systems which are described in Sec. 3. Results of the benchmark tests are shown in Sec. 4. Concluding remarks are provided in the last section.

2. Benchmark Codes

2.1 `gkvnl` code

One of the benchmark programs is the `GKV` code, which numerically solves the gyrokinetic equation for the one-body distribution function defined on the five-dimensional phase-space for tokamak and helical magnetic field configurations [4, 5]. A reduced version of the `GKV` code, `gkvnl`, is prepared for the benchmark test. The `GKV` code is implemented with the local flux tube model where the equilibrium quantities and their radial derivatives are assumed to be constant. In the flux tube coordinates, the gyrokinetic equations for perturbed quantities following the periodic boundary conditions for the radial (r) and field-line-label (α) directions are Fourier-transformed for r and α. Thus, we use five-dimensional arrays of double-precision complex numbers. The poloidal angle is measured along the field line. See Ref. [6] for more details of the flux tube model. The nonlinear advection term due to the electric-field ($\boldsymbol{E} \times \boldsymbol{B}$) drift motions of particles is, thus, calculated by the spectral method using a two-dimensional fast Fourier transform (FFT) method. The other three coordinates (the poloidal angle θ, the parallel velocity $v_\parallel$, and

author's e-mail: watanabe.tomohiko@nifs.ac.jp

Table 1 Summary of hardware and software systems employed in benchmark tests

System Used	Plasma Simulator	Earth Simulator	HPCx	SR11000	Franklin	Ranger
Number of nodes	5	640	160	128	9660	3936
Processor cores on a node	32	8	16	16	2	16
Total cores	160	5120	2560	2048	19320	62976
Processor	SX-7 (vector)	ES (vector)	Power5 (1.5 GHz)	Power5+ (2.3 GHz)	Opteron (dual core 2.6 GHz)	Opteron (quad core 2.0 GHz)
Processor core peak performance (GFlops)	8.8	8.0	6.0	9.2	5.2	8.0
Theoretical peak performance (TFlops)	1.41	40	15.36	18.84	101.5	503
Memory on a node (GB)	256	16	32	128	4	32
Memory bandwidth per core (GB/s)	35.3	32.0	†2.4	‡8.5/4.3	5.3	2.65
Interconnect (GB/s)	8×2	12.3×2	2.24×2	12×2	7.6	1 (*P-to-P)
Interconnect method	crossbar	crossbar	**HPS / IBM	crossbar	3D torus	fat tree
Compiler	Fortran90/ SX	Fortran90/ ES	XL Fortran/ 10.1	Fortran90/ Hitachi	PGI/7.1	PGI/7.1
FFT library	ASL/SX R19.0	ASL/ES R18.0		MATRIX	fftw/3.1	fftw/3.1

†Result of the STREAM benchmark test

‡Bandwidth for data input/output to the memory controller

*P-to-P: Point-to-Point

**HPS: High Performance Switch

the magnetic moment μ) are discretized by numerical grids on which finite-difference methods are employed.

The `gkvnl` code is written in Fortran 90 [7], and calls FFT library subroutines optimized for each computer system (see Table 1). To achieve a high efficiency in massively parallel computations, the FFT operation is confined in a single MPI process. The three-dimensional domain of $(\theta, v_\parallel, \mu)$ is decomposed into $M_\theta M_v M_\mu$ sub-domains, where M_θ, M_v, and M_μ represent the number of decompositions in the θ, $v_\parallel$, and μ directions, respectively. The total number of MPI processes (M_p) is $M_\mathrm{p} = M_\theta M_v M_\mu$. The `gkvnl` code is optimized for efficient hybrid operations of vector, thread parallel, and MPI parallel processes on the Plasma Simulator and Earth Simulator systems. The standard MPI parallelization is employed for the other machines, where each processor core is assigned to a single MPI process.

We have tested two problems of different sizes, which are referred to as the large- and small-sized cases. The array size of the main variable (double-precision complex number) is $(85, 169, 320, 128, 64)$ and $(85, 169, 160, 64, 32)$ for the large and small cases, respectively. In the present benchmark test, the `gkvnl` code is executed for 20 simulation time steps, and the wall-clock time, not including the initial setting, is measured by calling the `mpi_wtime` subroutine. For saving the computational resources, however, the execution on Franklin at the National Energy Research Scientific Computing Center (NERSC) is stopped at the 10th time step, and the measured wall-clock time is doubled.

2.2 `ep-mhd5` code

The second benchmark program, `ep-mhd5`, is based on the `MEGA` code [8–10], which can simulate self-consistent interactions between an MHD fluid and energetic particles. In the `ep-mhd5` code, the MHD equations are coupled with the motions of energetic particles through the current density perpendicular to the magnetic field, and are solved with the 4th-order finite difference method. The time evolution of the energetic particles is described by the drift-kinetic equations and is simulated with the particle-in-cell (PIC) method. The 4th-order Runge-Kutta method is employed for time integration.

The coordinate system consists of the rotating helical coordinates (u^1, u^2, u^3), where u^1 and u^2 are orthogonal in the poloidal plane and u^3 is the toroidal angle. The directions of u^1 and u^2 rotate depending on u^3. This coordinate system is useful for simulating helical plasmas. The `ep-mhd5` code is parallelized using MPI with three-dimensional domain decomposition. The three-dimensional domain (u^1, u^2, u^3) is decomposed into $M_1 M_2 M_3$ sub-domains, where M_1, M_2, and M_3 represent the numbers of decompositions in the u^1, u^2, and u^3 directions, respectively. The total number of MPI processes (M_p) is $M_\mathrm{p} = M_1 M_2 M_3$. The spatial distribution of the energetic-particles is also decomposed into sub-domains. When particles move to adjacent sub-domains, the parti-

cle information is transferred from the old to the new sub-domains through MPI communication.

We have tested two problems with different sizes, large- and small-sized cases. The numbers of the (u^1, u^2, u^3) grids are (1024, 1024, 1280) and (512, 512, 640) for the large and small cases, respectively. The number of computational energetic-particles is the same as the number of grids. In the present benchmark test, the `ep-mhd5` code is executed for 10 simulation time steps, and the wall-clock time, not including the initial setting, is measured by calling the `mpi_wtime` subroutine.

In a typical test run on the Plasma Simulator (SX-7/160M5), about 36% of the total time is used for calculations and communications of the MHD part. Particle pushing and gathering account for 25% and 37% of the total cost, respectively. For the MPI communication time, the MHD part dominates over the energetic-particle part with the present ratio of numbers of grid points and computational energetic-particles. For a larger number of MPI processes, in general, the communication costs of transferring the MHD data in the domain-boundary regions to the nearest neighbors will increase, which leads to a higher percentage of time consumed in the MHD part. However, as long as a good parallel scaling remains, the details of the total time elapsed remain unchanged.

3. Benchmark Environment

The benchmark tests are conducted on six supercomputer sites as summarized in Table 1. Main features of each system are briefly reviewed below.

The Plasma Simulator system (SX-7/160M5) [11] at NIFS consists of five computation nodes. Each node has 32 vector processors and 256 GB of shared memory. The theoretical peak performance is 1.4 TFlops. Because of the memory limit, benchmark tests on the Plasma Simulator are carried out only for the small cases of `gkvnl` and `ep-mhd5`. We use results from the test runs on the Plasma Simulator system as a reference for comparison.

The Earth Simulator [12] at the Japan Agency for Marine-Earth Science and Technology (JAMSTEC) has 640 computation nodes, delivering a theoretical peak performance of 40 TFlops. Each node has eight vector processors and 16 GB of shared memory. Processor elements are similar to those of SX-7. All nodes are connected by a single-stage crossbar network.

The HPCx system [13] in the United Kingdom utilizes 160 nodes of IBM p5-575, which have a peak performance of 15.36 TFlops. Each node has 16 cores of Power5 (1.5 GHz) processors and 32 GB of shared memory. Computation nodes are connected by the IBM High Performance Switch (HPS). The HPCx system is used for benchmark tests of `ep-mhd5` for the small-sized case.

We also used the Hitachi SR11000 system [14] at the University of Tokyo. It consists of 128 computation nodes with 16 cores of Power5+ processors (2.3 GHz) which provide a theoretical peak performance of 18.84 TFlops. The shared memory on each node is 128 GB. The internode connection is built of a crossbar network. On the SR11000 system, we have conducted benchmark tests of `gkvnl` for the small-sized case.

The Franklin system [15] at NERSC consists of 9660 dual-core Opteron processors (while it has been upgraded recently to quad-core processors). The peak performance exceeds 100 TFlops. Each node has a dual-core Opteron processor (2.6 GHz) with 4 GB of memory. An internode connection of 7.6 GB/s is achieved by the SeaStar2 routing and communication chip. The network topology is a three-dimensional torus.

The Ranger system [16] of the Texas Advanced Computing Center (TACC) at the University of Texas at Austin is a huge cluster system having 15,774 quad-core Opteron processors (2.0 GHz). Four processors, each of which has 8 GB of memory, are connected via Hyper Transport in a computation node. The total peak performance is 503 TFlops, while 16384 processor cores are currently available for a single job.

4. Benchmark Results

First, we have conducted small-sized runs of the `gkvnl` and `ep-mhd5` codes on the Plasma Simulator. The small-sized runs of `gkvnl` require a total memory size of 1058 GB with 20 MPI processes and eight parallel threads, and it increases with the number of parallel processes because of added boundary regions for each sub-domain and working area. The small-sized case of `ep-mhd5` used 400 GB memory with 160 MPI processes. The large-sized runs require a memory that is about eight times the memory necessary for the small-sized case. Because of the memory limit, only small-sized runs can be conducted on Plasma Simulator, HPCx, and SR11000. On the Earth Simulator, the large-sized case for the `ep-mhd5` code can be tested using 256 nodes with 4 TB of memory space, whereas only the small-sized case can be benchmarked for the `gkvnl` code. In the following benchmark tests, the same source codes are employed for all platforms, and we used the compile options, as summarized in Table 2.

The benchmark results on the Plasma Simulator are used as the reference for comparison in Figs. 1-4. The measured wall-clock times of the small-sized runs are 487.91 s and 49.60 s for `gkvnl` and `ep-mhd5`, respectively. The effective performances are 456.8 GFlops and 310.7 GFlops, which correspond to 32.4% and 22%, respectively, of the theoretical peak performance.

Results of the benchmark tests for the small-sized case of the `gkvnl` code are summarized in Fig. 1, where the speed-up factor (S_s) with respect to the Plasma Simulator is plotted versus the number of processor cores. The ratio of the result from the Plasma Simulator ($\tau_P = 487.91$ s) and the measured wall-clock time (t_s) defines S_s such that

$$S_s = \tau_P / t_s \,. \tag{1}$$

Table 2 Compile options used for the benchmark tests.

	`gkvnl`	`ep-mhd5`
Plasma Simulator	-gmalloc -Pauto -Wf"-reserve=8"	-gmalloc -Pauto -Wf"-reserve=1"
Earth Simulator	-gmalloc -Pauto -Wf"-reserve=8"	-gmalloc -Pauto -Wf"-reserve=1"
HPCx		-q64 -O5 -qcache=auto -qarch=pwr5 -qtune=pwr5
SR11000	-64 -model=K1 -apad=8192:896:ALL -Oss -noparallel -pvfunc=3	
Franklin	-fastsse	-fastsse
Ranger	-O3	-O3

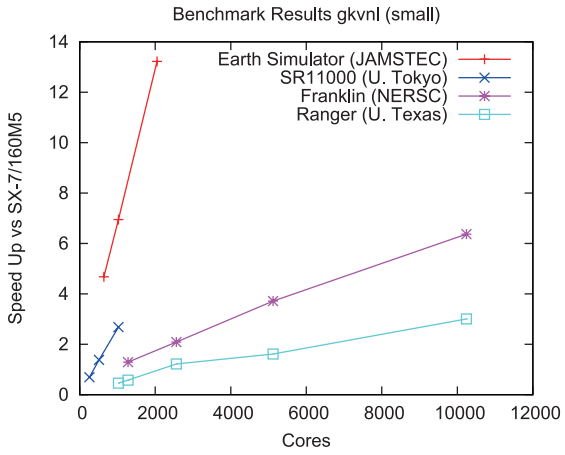

Fig. 1 Results of the benchmark test, `gkvnl` (small size), representing speed-up factors with respect to the effective performance of the Plasma Simulator (SX-7/160M5), which records 456.8 GFlops. The horizontal axis shows the number of processor cores.

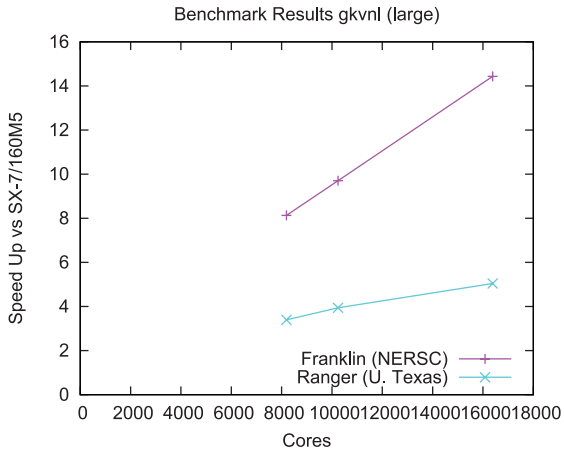

Fig. 2 Results of the benchmark test, `gkvnl` (large size), representing speed-up factors with respect to the effective performance of the Plasma Simulator (SX-7/160M5).

The speed-up factor increases with the number of processor cores for all machines. The result from 256 nodes of the Earth Simulator records 13.2 times faster operation than for the Plasma Simulator. It is also remarkable that the performance of Franklin and Ranger continues to increase over 10,000 cores. Speed-up factors of $S_s = 6.3$ and 3.0 are obtained for 10,240 cores of Franklin and Ranger, respectively. The SR11000 system of 1024 cores shows larger S_s values than those for 1280 cores of Franklin and Ranger because of the higher peak performance of the processor core and the wider memory bandwidth.

The benchmark results for the large case of the `gkvnl` code executed on Franklin and Ranger are shown in Fig. 2 in terms of the speed-up factor,

$$S_l = 8\tau_P/t_l \,, \tag{2}$$

where t_l is the measured wall-clock time for the large-sized case. In the definition of S_l, we use τ_P which is the same as that in Eq. (1). The factor 8 in Eq. (2) arises from the difference in the problem-size between the large and small cases. As seen in Fig. 2, the performance scaling over 10,000 cores is improved both for Franklin and Ranger. In the former, the speed-up of performance for 16,384 cores reaches $S_l = 14.4$ which corresponds to 6.4 TFlops.

Benchmark results of the `ep-mhd5` code for the small and large cases are shown in Figs. 3 and 4, respectively, where the speed-up factors are calculated from Eqs. (1) and (2). In the small-sized case, the speed-up factor of the Earth Simulator ($S_s = 11.1$) is slightly lower than that for the `gkvnl` code ($S_s = 13.2$). Results from Franklin and Ranger show better scaling than those obtained for the small-sized case of `gkvnl`. Speed-up factors of $S_s = 6.9$ and 3.4 are observed for 10,240 cores of Ranger and Franklin, respectively, while degradation of the parallelization efficiency is also found for a large number of cores. For a small number of cores (< 1280), HPCx shows a speed-up factor about 15% higher than that of Franklin, which is consistent with the difference in processor core peak performance.

The performance scaling of Franklin is improved for the large-sized case of `ep-mhd5` where S_l exceeds 11. The effective performance of Ranger, however, saturates for 16,384 cores. The difference in the parallelization scaling for large numbers of cores reflects the effective speed

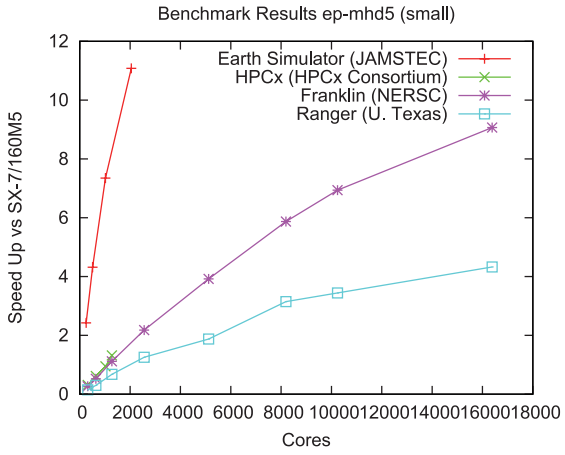

Fig. 3 Results of the benchmark test, `ep-mhd` (small size), showing speed-up factors with respect to the effective performance of the Plasma Simulator (SX-7/160M5), which records 310.7 GFlops.

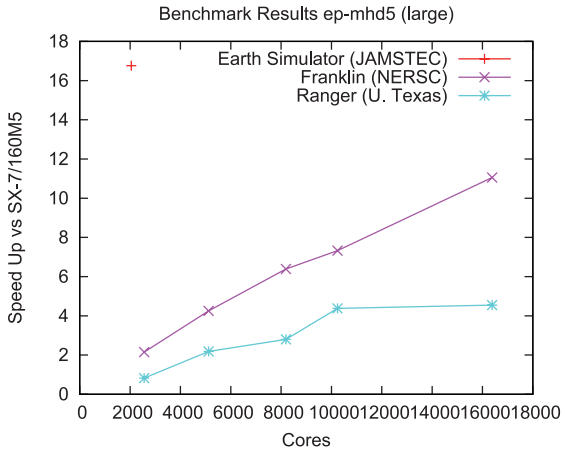

Fig. 4 Results of the benchmark test, `ep-mhd` (large size), showing speed-up factors with respect to the effective performance of the Plasma Simulator (SX-7/160M5).

of the internode connection. The benchmark test on 2048 cores of the Earth Simulator for the large case of `ep-mhd5` records a speed-up factor of 16.8, which corresponds to an effective performance of 5.2 TFlops.

Figure 5 summarized the effective performance per processor core obtained from the small-sized runs of the `gkvnl` and `ep-mhd5` codes with 1280 cores (but with 1024 cores for SR11000). The ratio of the effective core performances of Franklin and Ranger is about 2.23 for `gkvnl` and 1.65 for `ep-mhd5`. This is in contrast to the higher core peak performance of Ranger. Since both the codes show good linear scaling for 1280 cores, the difference in effective performance between the two systems is considered to arise from the memory bandwidth and/or latency. The result also suggests that the `gkvnl` code requires a higher

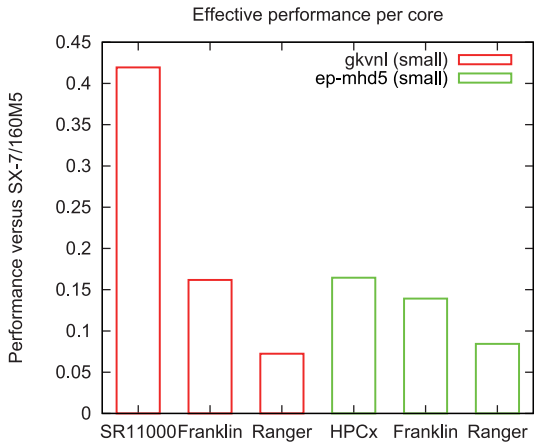

Fig. 5 Effective performance per processor core obtained from the small-sized runs with 1280 cores (but with 1024 cores for SR11000). The vertical axis is normalized by the effective performance of a single processor of the Plasma Simulator (SX-7/160M5).

memory throughput than `ep-mhd5`.

5. Concluding Remarks

Benchmark tests of `gkvnl` and `ep-mhd5`, the two major fusion plasma simulation codes developed at NIFS, are conducted on present-day parallel supercomputer systems. Both the codes, which were originally developed on a vector-parallel computer system, the Plasma Simulator (SX-7/160M5), achieve their high performance on the Earth Simulator, which has a similar architecture. The speed-up factors with respect to the Plasma Simulator exceed 13 and 11 for the small-sized test cases of `gkvnl` and `ep-mhd5`. The effective performance of the codes also shows good scaling on massively parallel computers with commodity processors. The effective performance per processor core of SR11000 for the `gkvnl` code is more than 40% of that for the SX-7 vector processor. New massively parallel computer systems with Opteron processors, Franklin and Ranger, enable us to use MPI processes over 10,000 cores. Even for a large number of parallel processes with an increasing cost of internode communications, the benchmark results of the `gkvnl` code show continuous growth of the effective performance for both Franklin and Ranger. The speed-up factor on Franklin is more than 14 for the large-sized case of `gkvnl`. The `ep-mhd5` code also exhibits a good performance on Franklin and Ranger, while growth of the scaling for the large-sized case saturates at 16,384 cores of Ranger due to increasing internode communications.

The present benchmark tests confirm that the two large-scale simulation codes for magnetic fusion plasmas can perform efficient computation on the state-of-the-art massively parallel supercomputer systems. As seen in

comparisons of Franklin and Ranger, the memory throughput and internode-connection speed are important issues for the `gkvnl` and `ep-mhd5` codes. For further improvement of performance in the near future, these concerns may be more crucial, since the number of processor cores on a chip will increase with relatively smaller enhancement in the memory throughput and the internode connection. Thus, it would be necessary to develop programming techniques that can optimize the codes in many-core systems, or to find new simulation algorithms or models that makes smaller demands on memory throughput and internode communications.

Acknowledgments

For providing computational resources that have contributed to the research results reported within this paper, the authors acknowledge the National Institute for Fusion Science (NIFS), the Institute for Fusion Studies, the University of Texas at Austin, the Earth Simulator Center (ESC) at Japan Agency for Marine-Earth Science and Technology, the Information Technology Center at the University of Tokyo, the HPCx Consortium, the National Energy Research Scientific Computing Center (NERSC), and the Texas Advanced Computing Center (TACC) at the University of Texas at Austin.

Some of the work carried out on Japanese site is supported in part by grants-in-aid from the Ministry of Education, Culture, Sports, Science and Technology (No.17360445 and 20340165), in part by the National Institute for Fusion Science (NIFS) Collaborative Research Program (NIFS05KKMT001, NIFS06KTAT025, NIFS06KTAT038, NIFS08KTAL006, and NIFS08KTAL010).

Some of the work carried out on US site is supported in part by the National Science Foundation (NSF) (No.ATM-0638480), and in part by the Department of Energy's Scientific Discovery through Advanced Computing (SciDAC) program (No.DE-FC02-08ER54961).

This US-Japan research collaboration is also supported by the Joint Institute for Fusion Theory program, and in part by National Institutes of Natural Sciences (NINS) under the project Formation of International Network for Scientific Collaborations (KEIN1004).

A part of this work made use of the facilities of HPCx, UK's national high-performance computing service, which is provided by EPCC at the University of Edinburgh and by CCLRC Daresbury Laboratory, and funded by the Office of Science and Technology through the EPSRC's High End Computing Program.

[1] T. Hayashi *et al*, J. Plasma Fusion Res. **79**, 464 (2003).
[2] W. Tang, Phys. Plasmas **9**, 1856 (2002).
[3] W. Gropp, E. Lusk and A. Skjellum, *Using MPI: portal parallel programing with the message-passing interface*, Second Edition (The MIT Press, Cambridge, Massachusetts, USA, 1999).
[4] T.-H. Watanabe and H. Sugama, Nucl. Fusion **46**, 24 (2006).
[5] T.-H. Watanabe, H. Sugama and S. Ferrando-Margalet, Nucl. Fusion **47**, 1383 (2007).
[6] M.A. Beer, S.C. Cowley and G.W. Hammett, Phys. Plasmas **2**, 2687 (1995).
[7] T.M.R. Ellis, I.R. Philips and T.M. Lahey, *Fortran 90 programming* (Addison-Wesley, Wokingham, U.K., 1994).
[8] Y. Todo and T. Sato, Phys. Plasmas **5**, 1321 (1998).
[9] Y. Todo, K. Shinohara, M. Takechi and M. Ishikawa, Phys. Plasmas **12**, 012503 (2005).
[10] Y. Todo, N. Nakajima, K. Shinohara, M. Takechi, M. Ishikawa and S. Yamamoto, in Fusion Energy 2004 (Proc. 20th Int. Conf. Vilamoura, 2004) IAEA, Vienna, TH/3-1Ra.
[11] http://www.nec.co.jp/hpc/sx7/index.html
[12] http://www.jamstec.go.jp/esc/
[13] http://www.hpcx.ac.uk/
[14] http://www.cc.u-tokyo.ac.jp/service/intro/index.html
[15] http://www.nersc.gov/nusers/systems/franklin/
[16] http://www.tacc.utexas.edu/resources/hpcsystems/