Optimization of stencil-based fusion kernels on Tera-flops many-core architectures

Y. ASAHI
Japan Atomic Energy Agency
Wakashiba 178-4, Kashiwa, Chiba
Japan
+81-4-7135-2346
asahi.yuichi@jaea.go.jp

G. Latu
CEA, Cadarache
F-13108, Saint-Paul-Lez-Durance
France
+33-442-25-61-19
guillaume.latu@cea.fr

T. Ina
Japan Atomic Energy Agency
Wakashiba 178-4, Kashiwa, Chiba
Japan
+81-4-7135-2356
takuya.ina@jaea.go.jp

Y. Idomura
Japan Atomic Energy Agency
Wakashiba 178-4, Kashiwa, Chiba
Japan
+81-4-7135-2454
idomura.yasuhiro@jaea.go.jp

V. Grandgirard
CEA, Cadarache
F-13108, Saint-Paul-Lez-Durance
France
+33-442-25-61-19
virginie.grandgirard@cea.fr

X. Garbet
CEA, Cadarache
F-13108, Saint-Paul-Lez-Durance
France
+33-442-25-61-29
xavier.garbet@cea.fr

ABSTRACT

We present the optimization of kernels from fusion plasma codes, GYSELA and GT5D, on Tera-flops many-core architectures including accelerators (Xeon Phi, TeslaK20X), and CPUs (FX100). Through the optimization, we found that the structure of array (SoA) style implementation is effective for SIMD operations on all architectures, and high cache locality, which is achieved in GYSELA, is of critical importance on accelerators. Also, the OpenMP dynamic scheduling is effective to overcome the drawbacks of in-order instruction processors such as Xeon Phi.

Categories and Subject Descriptors

[**Programming Languages**]: F77, F90 and CUDA Fortran; [**Computing methodologies**]: Massively parallel computations on many-core architecture; [**Applied computing**]: Plasma physics

General Terms

Algorithms, Performance

Keywords

Plasma turbulence, Parallel performance, Many-core architecture

1. INTRODUCTION

In computational fusion science, we need more computing power to enable first principle simulations of the next generation fusion device ITER [1] including multi-species ions and electrons dynamics. This motivates us to develop new optimization techniques of fusion plasma codes on Tera-flops many-core architectures such as Intel Xeon Phi 5110P (Phi) [2], Nvidia TeslaK20X (K20X) [3] and Fujitsu FX100 (FX100) [4], which are candidates for future Exa-scale architectures (see table 1). FX100 is classified as a commodity multi-core CPU, while Phi and K20X are accelerators focusing on computation. As a reference, we employed a single socket Intel Xeon Sandy Bridge EP (SB).

We address two types of fusion plasma codes, GYSELA [5] and GT5D [6], which compute evolutions of plasma distribution functions in 5D phase space (x, y, z, v, μ). The most time consuming part is computation of a 4D convective operator, which is a partial difference operator in (x, y, z, v). GYSELA and GT5D compute the operator with a semi-Lagrangian scheme and a finite different scheme, respectively. We extract kernels from these codes to evaluate their performances on different kinds of many-core architectures. In this study, we focus on computation rather than communication, and work on serial processing kernels (without MPI). As they have different arithmetic intensity and memory access patterns, they show different characteristics and have to be optimized in a different manner. In the following, we present the optimization of these kernels on architectures in table 1 and discuss their arithmetic properties.

Table 1 Machine environment

Processor	SB	Phi	K20X	FX100
Cores	8	60	896	32+2
Memory [GB]	64	8	6	32
Cache [MB]	20	0.5 x 60	1.5	24
Peak Flops (DP) [GFlops]	172.8	1010	1310	1000
Peak B/W [GB/s]	51.2	320	250	480
SIMD width	4(AVX)	8	-	4
B/F ratio	0.3	0.3	0.19	0.5

2. GYSELA and GT5D kernels

We describe GYSELA and GT5D kernels as summarized in table 2.

2.1 The GYSELA kernel

GYSELA kernel computes 2D cubic spline interpolation extracted from the 4D convection operator. Since the 4D convection operator is split into 1D+1D+2D parts, the 2D part is most expensive [7]. This kernel is “compute intensive” whose Byte/Flop ratio is 0.37. In Ref. [7], GYSELA kernel was optimized on SB and Phi. The strategy was to keep good vectorization, fine grain parallelism, data alignment, effective cache usage, prefetching and loop splitting. The optimized GYSELA kernel gave 78.01 GFlops (45.1% to peak) on SB (single socket).

2.2 The GT5D kernel

GT5D kernel computes the 4D convection operator with a 4th order finite difference (17 stencils) [6]. This kernel is "memory bound" whose Byte/Flop is 1. The memory access is dominated by finite difference computation with respect to the outermost loop index. In Ref [8], GT5D kernel was optimized for multi-core CPUs by using OpenMP cyclic loop-decomposition, which enables a thread to reuse the data loaded on a shared cache by adjacent threads. This technique improves Byte/Flop to 0.5 so that the optimized GT5D kernel gives 28.03 GFlops (16.22%) on SB (single socket).

Table 2 Kernel information

Kernel	GYSELA	GT5D
Scheme	2D Cubic Spline [8]	4D FD (17stencil) [7]
Problem size	(128, 72, 52, 201)	(128, 16, 32, 32)
Memory access	1 load/store + 2 load	1 load/store + 6 load
Flop	87	66
Byte / Flop	0.37	1

3. Optimization

3.1 Optimization of GYSELA kernel

Although GYSELA kernel was almost fully optimized [7], we further optimize the kernel with respect to rearrangement of data structure to SoA style for sequential memory access and OpenMP dynamic loop-decomposition. The former is essential for SIMD load/store. The latter improves the efficiency of pipeline usage in many thread execution, since Phi is in-order instruction processor. Table 3 shows improvements with the optimization, where we find 344 GFlops (34%). For the Phi, the performance is significantly affected by Intel compiler version, where the best performance is obtained with version 13.1.3. The performance on Phi, K20X and FX100 is shown in Figure 1. The GPU-version is fundamentally the same as Phi-version which gives 389 GFlops (29.7%). A key ingredient for the good performance on accelerators is to keep the spline coefficient on the local-cache.

Table 3 Optimization of GYSELA kernel on Phi

Graphics	Original	SIMD	Dynamic
GFLOPS	250	310	344
Relative to peek (%)	25	31	34

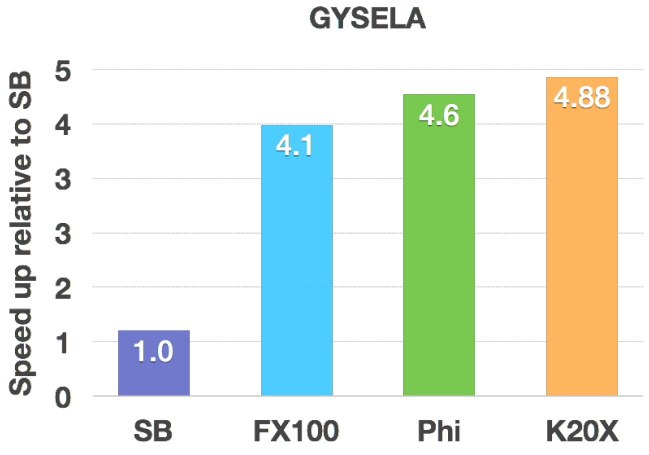

Figure 1. Performance of GYSELA kernel on Sandy Bridge (single socket), Xeon Phi, TeslaK20X and FX100. Speed up relative to SB is shown.

3.2 Optimization of GT5D kernel

The performance on Phi, K20X and FX100 is shown in Figure 2. On FX100, the reuse of shared cache between adjacent threads works well. However, this technique seems ineffective on Phi due to large remote cache access latency [11], which would degrade arithmetic intensity. On K20X, the limited shared cache makes it difficult to apply this technique, and the outermost dimension requires memory access. In addition, further optimization is needed for GPU. Since the original GPU kernel [9] includes a divergent branch, we firstly modified the dimension computed by each thread. We then applied tiling to load data and halo region with coalesced data access [10]. Finally, we obtained 95.58 GFlops (7.3%).

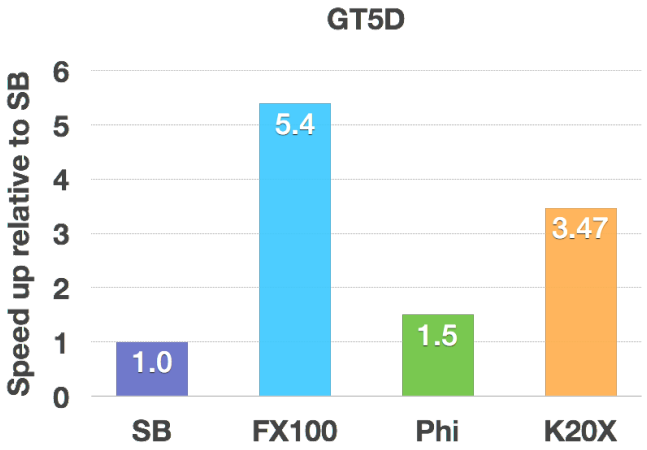

Figure 2. Performance of GT5D kernel on Sandy Bridge (single socket), Xeon Phi, TeslaK20X and FX100. Speed up relative to SB is shown.

4. ACKNOWLEDGMENTS

This work is supported by the HPCI Strategic Programs Field No. 4, the CEA-JAEA collaboration program. The computation is carried out on Helios (Iferc), FX100 (NIFS), Tsubame2.5 (Tokyo Tech).

5. REFERENCES

[1] ITER. https://www.iter.org

[2] INTEL. Intel xeon phi coprocessor inst. set archi. ref. manual. https://software.intel.com/en-us/mic-developer.

[3] Nvidia. TESLA K20X GPU ACCELERATOR http://www.nvidia.co.jp/content/PDF/kepler/Tesla-K20X-BD-06397-001-v07.pdf

[4] Fujitsu. FUJITSU Supercomputer PRIMEHPC FX100 http://www.fujitsu.com/global/Images/primehpc-fx100-datasheet-en.pdf

[5] V. Grandgirard, et al., *Computing ITG turbulence with a full-f semi-lagrangian code*, Commun. Nonlinear Sci. Num. Simul., 13 81, 2008.

[6] Y. Idomura, et al., *Conservative global gyrokinetic toroidal full-f five-dimensional Vlasov simulation*., Comput. Phys. Commun. 179 391, 2008.

[7] G. Latu, M. Haefele, J. Bigot, V. Grandgirard, T. Cartier-Michaud and F.Rozar., *Evaluating kernels on Xeon Phi to accelerate Gysela application*. Submitted to proceedings of ESAIM, 2015.

[8] Y. Idomura, et al., *Communication-overlap techniques for improved strong scaling of gyrokinetic Eulerian code beyond 100k cores on the K-computer*, International Journal of High Performance Computing Applications 28, 73-86, 2014.

[9] N. Fujita, Hideo Nuga, Taisuke Boku, Yasuhiro Idomura, *Nuclear Fusion Simulation Code Optimization on GPU Clusters*, The 15th IEEE International Workshop on Parallel and Distributed Scientific and Engineering Computing, 2014.

[10] Marcin Krotkiewski and Marcin Dabrowski, *Efficient 3D stencil computations using CUDA*, Parallel Computing 39 (10) s 533-548, 2013.

[11] Jianbin Fang, Ana Lucia Varbanescu, Henk J. Sips, Lilun Zhang, Yonggang Che, and Chuanfu Xu, *An empirical study of Intel Xeon Phi*, CoRR, 2013.