Hybrid CPU + Xeon Phi implementation of the Particle-in-Cell method for plasma simulation

DOI: 10.14529/jsfi160301

Hybrid CPU + Xeon Phi implementation of the Particle-in-Cell method for plasma simulation

Iosif B. Meyerov[1], *Sergey I. Bastrakov*[1], *Igor A. Surmin*[1], *Alexey V. Bashinov*[2], *Evgeny S. Efimenko*[12], *Artem V. Korzhimanov*[12], *Alexander A. Muraviev*[12], *Arkady A. Gonoskov*[123]

This paper presents experimental results of Particle-in-Cell plasma simulation on a hybrid system with CPUs and Intel Xeon Phi coprocessors. We consider simulation of two relevant laser-driven particle acceleration regimes using the Particle-in-Cell code PICADOR. On a node of a cluster with 2 CPUs and 2 Xeon Phi coprocessors the hybrid CPU + Xeon Phi configuration allows to fully utilize the computational resources of the node. It outperforms both CPU-only and Xeon Phi-only configurations with the speedups between 1.36 x and 1.68 x.

Keywords: hybrid computing, Xeon Phi, Particle-in-Cell.

Introduction

Numerical simulation of plasmas is an important area of computational physics with numerous applications. The Particle-in-Cell method [1–3] is widely used for plasma simulation in ultrahigh fields. Large-scale 3D simulation requires the use of supercomputers. Currently, there are a number of Particle-in-Cell implementations capable of that, including OSIRIS [4], PIConGPU [5], VPIC [6], VLPL [7], WARP [8].

During the recent years there has been a continuous progress in utilization of accelerators, including GPUs [5, 9, 10] and Intel Xeon Phi coprocessors [11, 12]. Our code PICADOR has been previously ported and optimized for Xeon Phi coprocessors that led to 1.6–1.8 x speedup compared to an 8-core CPU on a benchmark problem [13]. Naturally, it is more challenging to efficiently utilize Xeon Phi on real applications that may have large output, significant imbalance, costly diagnostics, and other performance hindering factors. However, even obtaining the same performance on Xeon Phi as on a multicore CPU is beneficial in terms of hybrid CPU + Xeon Phi computing. It allows to fully utilize computational resources of heterogeneous machines with several CPUs and Xeon Phi coprocessors per node and theoretically may significantly outperform CPU-only and Xeon Phi-only configurations.

This paper presents our experience of solving two problems by Particle-in-Cell simulation using PICADOR on a hybrid CPU + Xeon Phi machine. These two problems have been selected as they are relevant for laser-plasma physics and are part of our current research involving PICADOR as a tool for numerical simulation. Thus, the problems considered present a realistic workload and, secondly, any speedup obtained due to Xeon Phi or hybrid CPU + Xeon Phi computing is beneficial for the current research done using the code. The paper is organized as follows. We briefly describe the method and our code PICADOR in section 1. Sections 2 and 3 are devoted to the physical problems we consider. Summary and conclusions are given in section 3.

[1]Lobachevsky State University of Nizhni Novgorod, Nizhni Novgorod, Russia
[2]Institute of Applied Physics, Russan Academy of Sciences, Nizhni Novgorod, Russia
[3]Chalmers University of Technology, Gteborg, Germany

1. Particle-in-Cell code PICADOR

1.1. Particle-in-Cell method

In this subsection we briefly describe the Particle-in-Cell method, a detailed description is given in [3].

The Particle-in-Cell method operates on field data and particle data. Values of electric and magnetic fields are defined on a spatial grid. A plasma is represented as an ensemble of particles, each with a charge, mass, position and momentum. Each particle used in simulation is in fact a macroparticle that represents a cloud of real particles. A particle form factor defines particle distribution inside the cloud. In this paper we use the cloud-in-cell form factor that corresponds to the uniform distribution in the cloud that has the same size as a cell of the spatial grid. A notable feature of the method is that particles do not interact with each other directly, instead each particle interacts with a set of nearby grid values, depending on the form factor.

The basic computational loop of the Particle-in-Cell method consists of the following stages. For each particle the Lorenz force is computed using interpolated values of the electromagnetic field and the particle momentum and position are updated. Grid values of the current created by particle movement are computed. Field interpolation and current deposition depend on the particle form factor being used. Finally, field values are updated by solving Maxwell's equations.

1.2. PICADOR code

PICADOR [13, 14] is a tool for 3D plasma simulation based on the Particle-in-Cell method. The code supports a rather standard set of numerical schemes for laser-plasma simulation: several widely used particle form factors, Boris particle pusher [15], Yee grid [16] and finite-difference time-domain field solver [17], charge-conserving current deposition [18, 19].

Parallel computing on cluster systems is organized using MPI. On the internode level we use a 3D rectilinear domain decomposition, with each MPI process handling a part of the simulation area. Each MPI process stores particles and grid values in the corresponding sub-area with guard cells. Data exchanges occurs only between neighbour sub-areas. On shared memory OpenMP is used to parallelize loops over particles and cells. Processing of particles in a cell is partially vectorized using a combination of compiler auto-vectorization and manually coded intrinsic functions, details are given in [13]. The basic computational scheme is given in fig. 1.

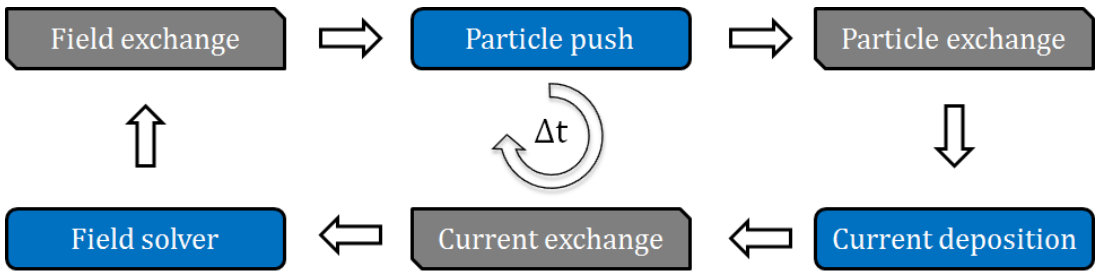

Figure 1. Computational scheme of the Particle-in-Cell method

PICADOR has been previously ported and optimized for Intel Xeon Phi coprocessors [13]. Applying standard optimization techniques such as improving memory locality, enhancing scaling efficiency and vectorization resulted in 1.6–1.8 x speedup on Xeon Phi 5110P relative to an 8-core Intel Xeon E5-2660 CPU on a benchmark problem with a uniform distribution of particles between threads and no MPI communication.

2. Hybrid simulation of wakefield laser pulse self-compression

In this section we consider simulation of a laser pulse self-compression due to wakefield excitation in the relativistic regime. A gas jet is irradiated by a femtosecond high intensity laser pulse. In the result of the interaction a generated wakefield may lead to laser pulse self-compression. This is a promising mechanism for generating ultra-short high-intensity laser pulses [20]. The simulation was performed using the following parameters: $256 \times 128 \times 128$ grid, 78.5 million particles, 7 674 time steps, cloud-in-cell particle form factor, charge-conserving Villasenor-Buneman current deposition scheme [18], double precision floating point arithmetic.

Computational experiments were performed on a node of the MVS-10P supercomputer at the Joint Supercomputer Center of RAS with 2 x Intel Xeon E5-2690 CPU (8 cores, 2.9 GHz, Hyperthreading enabled) and 2 x Intel Xeon Phi 7110 (61 cores, 1.053 GHz), Intel C++ Compiler 14.0.1, Intel MPI 4.1. We used a single MPI process per CPU and Xeon Phi, 2 OpenMP threads per core on CPU and 4 threads per core on Xeon Phi. Our previous results [13] show that this configuration of processes and threads is the best for PICADOR. This is probably due to the fact that during the most computationally intensive stages of the Particle-in-Cell method there are lots of independent subproblems that can be solved in parallel, even with 4 threads per core on Xeon Phi. Meanwhile, running several MPI processes on shared memory requires some additional communication, which could hinder overall performance. Run time on a node of MVS-10P in three configurations: CPU-only, Xeon Phi-only, and hybrid CPU + Xeon Phi is given in tab. 1.

Table 1. Run time of simulation of wakefield laser pulse self-compression on CPU, Xeon Phi, and CPU + Xeon Phi. Time is given in seconds. For the computational core a split into current deposition, particle push, and other operations is given

Stage	2 x CPU	2 x Xeon Phi	2 x CPU + 2 x Xeon Phi
Computational core overall	693.2	681.3	351.2
particle push	408.9	341.6	201.7
current deposition	253.5	314.4	129.6
other	30.8	25.3	19.8
MPI communication	21.1	148.5	144.2
Overall	714.3	829.8	495.3

Run time of the computational core on Xeon Phi is close to that of the CPU. There is some advantage in the particle push stage which is partially vectorized on Xeon Phi, but it is compensated by the non-vectorized current deposition stage. MPI communication on Xeon Phi is slower because some related operations are sequential and single-core performance of the coprosessor is inferior to the CPU. Overall, Xeon Phi is 1.16 x slower compared to the CPU. Nevertheless, utilizing the coprocessors allows to accelerate the simulation compared to the CPU by means of the hybrid CPU + Xeon Phi configuration. It yields 1.44 x and 1.68 x speedup compared to the CPU-only and Xeon Phi-only configurations, respectively.

3. Hybrid simulation of target normal sheath acceleration

In this section we consider a laser ion acceleration in the target normal sheath acceleration regime, which is a widely used approach to laser-driven particle acceleration [21]. A high-intensity

laser pulse heats electrons at the front surface of a target forming a sheath at its rear side. Ions of the target are accelerated from the rear side by the electron sheath. Surface grating is used on the irradiated side of the target to increase efficiency [22]. The hardware and numerical schemes were the same as in the previous section. Run time on a node of MVS-10P in CPU-only, Xeon Phi-only and CPU + Xeon Phi configurations is given in tab. 2.

Table 2. Run time of simulation of target normal sheath acceleration on CPU, Xeon Phi, and CPU + Xeon Phi. Time is given in seconds. For the computational core a split into current deposition, particle push, and other operations is given

Stage	2 x CPU	2 x Xeon Phi	2 x CPU + 2 x Xeon Phi
Computational core overall	3 012.6	2 137.0	1 687.3
particle push	1 888.8	1 214.7	1 004.3
current deposition	974.0	839.7	605.3
other	149.8	82.6	77.7
MPI communication	292.9	1 637.3	748.8
Overall	3 305.5	3 774.3	2 436.2

The simulation area was divided between MPI processes along the direction of the laser pulse to reduce the number of particles going from one process to another. Nevertheless, due to the complex dynamics occurring throughout the simulation, there is a massive migration of particles between processes. Therefore, a large amount of time is spent on MPI communication: 8.8% of the total run time on the CPU and 43.4% on the coprocessor. Due to this fact Xeon Phi is 1.14 x slower compared to the CPU, although the computational core is 1.41 x faster. The issue of the slow particle migration is partially alleviated in the hybrid configuration because of the lower amount of particles per process. The CPU + Xeon Phi combination outperforms the CPU-only and Xeon Phi-only runs by 1.36 x and 1.55 x, respectively.

Conclusions

This paper presents results of simulation of two problems on a node of the MVS-10P cluster using three configurations: 2 x CPUs, 2 x Xeon Phi coprocessors, 2 x CPUs + 2 x Xeon Phi. Both problems considered are not ideally suited for Xeon Phi, with the CPU slightly outperforming Xeon Phi by factors of 1.16 x and 1.14 x because of MPI communication time. Even in this case, it is possible to use the Xeon Phi coprocessors to accelerate simulation by fully utilizing the computational resources of a node in the hybrid CPU + Xeon Phi mode. The speedup of the hybrid configuration compared to the CPU-only and Xeon Phi-only modes is between 1.36 x and 1.68 x. Further progress can be made by reducing MPI communication time, particularly on Xeon Phi. A possible approach is to first process near-boundary particles and then overlap asynchronous particle transfers and processing the remaining particles. Our very first implementation of this idea shows some speedup of data transfers on Xeon Phi for the problem considered in section 2: 1.46 x on 2 x Xeon Phi and 1.35 x in the hybrid mode. However, a tradeoff is some increase in the particle push time, so that the overall speedup is 1.05 x. Further improvement of data transfers is a direction of future work.

The problems considered in this paper have a sufficiently uniform distribution of particles in the simulation area. Thus, they are rather suitable for the manycore architecture of Xeon Phi

with the standard OpenMP thread scheduling policies. However, there are important regimes involving electron-positron cascades [23, 24] with a small number of cells having more particles than the rest of the simulation area. Our first experiments show that in this case the utilization of standard approach results in significant thread workload imbalance on Xeon Phi that causes underwhelming performance. A parallelization scheme for such problems is under development.

The recently introduced Intel Xeon Phi products of the KNL generation look very promising due to a significant increase in the overall performance as well as in the single-core performance. Optimization of the code for KNL is a direction of future work.

This study was supported by the RFBR, project No. 15-37-21015.

References

1. Hockney R.W., Eastwood J.W. Computer simulation using particles. New York: McGraw-Hill. 1981.

2. Dawson J.M. Particle simulation of plasmas, Reviews of Modern Physics. 1983. 55 (2). 403–447. DOI: 10.1103/RevModPhys.55.403.

3. Birdsal C.K., Langdon A.B. Plasma physics via computer simulation. CRC Press. 2004.

4. Fonseca R.A., Vieira J., Fiuza F., Davidson A., Tsung F.S., Mori W.B., Silva L.O. Exploiting multi-scale parallelism for large scale numerical modelling of laser wakefield accelerators. Plasma Phys. Control. Fusion. 2013. 55 (12). 124011.

5. Burau H., Widera R., Honig W., Juckeland G., Debus A., Kluge T., Schramm U., Cowan T.E., Sauerbrey R., Bussmann M. PIConGPU: A Fully Relativistic Particle-in-Cell Code for a GPU Cluster. IEEE Transactions on Plasma Science. 2010. 38 (10). 2831–2839.

6. Bowers K.J., Albright B.J., Yin L., Bergen B., Kwan T.J.T. Ultrahigh performance three-dimensional electromagnetic relativistic kinetic plasma simulation. Physics of Plasmas. 2008. 15 (5). 055703.

7. Pukhov A. Three-dimensional electromagnetic relativistic particle-in-cell code VLPL (Virtual Laser Plasma Lab). Journal of Plasma Physics. 1999. 61 (3). 425–433.

8. Vay J.-L., Bruhwiler D.L., Geddes C.G.R., Fawley W.M., Martins S.F., Cary J.R., Cormier-Michel E., Cowan B., Fonseca R.A., Furman M.A., Lu W., Mori W.B., Silva L.O. Simulating relativistic beam and plasma systems using an optimal boosted frame. Journal of Physics: Conference Series. 2009. 180 (1). 012006.

9. Kong X., Huang M.C., Ren C., Decyk V.K. Particle-in-cell simulations with charge-conserving current deposition on graphic processing units. Journal of Computational Physics. 2011. 230 (4). 1676–1685. DOI: 10.1016/j.jcp.2010.11.032.

10. Decyk V.K., Singh T.V. Particle-in-Cell algorithms for emerging computer architectures. Computer Physics Communications. 2014. 185 (3). 708–719. DOI: 10.1016/j.cpc.2013.10.013.

11. Nakashima H. Manycore challenge in particle-in-cell simulation: how to exploit 1 TFlops peak performance for simulation codes with irregular computation. Computers and Electrical Engineering. 2015. 46. 81–94. DOI: 10.1016/j.compeleceng.2015.03.010.

12. Glinsky B.M., Kulikov I.M., Snytnikov A.V., Romanenko A.A., Chernykh I.G., Vshivkov V.A. Co-design of parallel numerical methods for plasma physics and astrophysics. Supercomputing Frontiers and Innovations. 2014. 1 (3). 88–98.

13. Surmin I.A., Bastrakov S.I., Efimenko E.S., Gonoskov A.A., Korzhimanov A.V., Meyerov I.B. Particle-in-Cell laser-plasma simulation on Xeon Phi coprocessors. Computer Physics Communications. 2016. 202. 204–210. DOI: 10.1016/j.cpc.2016.02.004.

14. Bastrakov S., Donchenko R., Gonoskov A., Efimenko E., Malyshev A., Meyerov I., Surmin I. Particle-in-cell plasma simulation on heterogeneous cluster systems. Journal of Computational Science. 2012. 3. 474–479.

15. Boris J.P. Relativistic plasma simulation-optimization of a hybrid code. Proceedings of the 4th Conference on Numerical Simulation of Plasmas. 1970. 3–67.

16. Yee K. Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media. IEEE Transactions on Antennas and Propagation. 1966. 14 (3). 302–307. DOI: 10.1109/TAP.1966.1138693.

17. Taflove A. Computational electrodynamics: the finite-difference time-domain method. London: Artech House. 1995.

18. Villasenor J., Buneman O. Rigorous charge conservation for local electromagnetic field solvers. Computer Physics Communications. 1992. 69. 306–316. DOI: 10.1016/0010-4655(92)90169-Y.

19. Esirkepov T.Zh. Exact charge conservation scheme for Particle-in-Cell simulation with an arbitrary form-factor. Computer Physics Communications. 2011. 135. 144–153. DOI: 10.1016/S0010-4655(00)00228-9.

20. Pipahl A., Anashkina E.A., Toncian M., Toncian T., Skobelev S.A., Bashinov A.V., Gonoskov A.A., Willi O., Kim A.V. High-intensity few-cycle laser-pulse generation by the plasma-wakefield self-compression effect. Physical Review E. 2013. 87. 033104.

21. Wilks S.C., Langdon A.B., Cowan T.E., Roth M., Singh M., Hatchett S., Key M.H., Pennington D., MacKinnon A., Snavely R.A. Energetic proton generation in ultra-intense lasersolid interactions. Physics of Plasmas. 2001. 8 (2). 542–549.

22. Bychenkov V.Y., Brantov A.V., Govras E.A., Kovalev V.F. Laser acceleration of ions: recent results and prospects for applications. Physics-Uspekhi. 2015. 58 (1). 71–81.

23. Bell A.R., Kirk J.G. Possibility of Prolific Pair Production with High-Power Lasers. Physical Review Letters. 2008. 101 (20). 200403. DOI: 10.1103/PhysRevLett.101.200403.

24. Gonoskov A., Bastrakov S., Efimenko E., Ilderton A., Marklund M., Meyerov I., Muraviev A., Sergeev A., Surmin I., Wallin E. Extended Particle-in-Cell Schemes for Physics in Ultrastrong Laser Fields: Review and Developments. Physical Review E. 2015. 92 (2). 023305. DOI: 10.1103/PhysRevE.92.039903.