IMPLEMENTATION OF THE MODERN PLASMA SIMULATION CODES VIA PIC METHOD FOR PARALLEL COMPUTING SYSTEMS

D.I. Dadyka, I.O. Anisimov

Taras Shevchenko National University of Kyiv, Kiev, Ukraine

E-mail: d.dadyka@gmail.com

The comparison of common open source software for the simulation of plasma via particles-in-cell (PIC) method using parallel computing systems is presented. The problems of field equation solving, load balancing, general-purpose computing on graphics processing units are considered. All the reviewed programs have some disadvantages, in particular associated with the used field solving methods, data caching and with lack of the adaptive grids support. The approach for cache misses minimizing based on the particles sorting is brought forward. The algorithm for effective Poisson solving is proposed.

PACS: 02.60.Pn, 52.65.Rr, 52.80.Tn

INTRODUCTION

Computer simulation has become one of the important research methods in plasma physics. The particle-in-cell (PIC) method is one of the most common methods for such simulation. However, the existing packages are imperfect. In particular, the used parallelization schemes are not optimal. The purpose of this work is the investigation of the possible ways for improving these packages.

The implementation of modern open source simulation programs via PIC method is considered. The review includes the programs XOOPIC [1], PICCANTE [2], PSC [3, 4], PIConGPU [5] and CPIC [6]. All these programs support the operation via the distributed parallel computing systems with MPI (message passing interface). The program codes can be downloaded from "github" and compiled by "GNU make" utility. The comparison of packages is shown in Table 1.

1. CACHE MISSES AND PIC SIMULATION

The most modern computer systems' architectures use the caching. When CPU (central processor unit) accesses to the main memory, some data array is copied from the main memory into the faster cache memory. Thus, access to the following memory cells will be faster because the data will be cached. But cashing requires a consistent memory access addresses [7]. In typical PIC code we need to integrate the equations of motion:

$$m\frac{d\vec{v}_i}{dt} = q\vec{E}_k\,, \tag{1}$$

$$\frac{d\vec{x}_i}{dt} = \vec{v}_i\,, \tag{2}$$

where $\vec{v}_i$ is the velocity of current particle i, $\vec{E}_k$ is the field in the grid cell k where the particle is placed, k is determined by the coordinates of particle $\vec{x}_i$. The relationship between i and k is random. Fig. 1,a illustrates the memory accesses scheme. Cache misses happen very often, so low productivity is obtained.

We propose to make the sorting of particles with a certain periodicity, so i becomes dependent on k. Particles can't move to more than one cell for one time step, so sorting may not be frequent. We propose to use space-filling Peano curve for mapping 2D space at 1D one by $PC(x, y)=n$ formula which binds Peano curve point coordinates x, y in 2D space and point number n. For sorting we propose to use the smoothsort algorithm [8] because it comes closer to $O(n)$ time if the input is already sorted to some degree.

Table 1

PIC packages comparison. Abbreviations: MG – multi grid solver, CG – conjugate gradient solver, ADI – alternating direction implicit solver, FDTD – finite difference time domain solver, LB – load balancing support, GPGPU – general purpose computing on graphics processing unit

Package	Year	LB	GPGPU	Dimensions	Field solver	Interactions
XOOPIC	1996	-	-	2D	MG, CG, ADI	Monte-Carlo
PSC	2012	+	+	3D	FDTD	Monte-Carlo
PICCANTE	2014	-	-	3D	FDTD	-
PIConGPU	2013	-	+	3D	FDTD	Thomas-Fermi ionization
CPIC	2014	-	-	3D	MG	-

Tests' result for moving 10^6 particles in 2D space without sorting and for sorting every 20 steps is presented in Table 2. Sorting time is very small because the particles' array is sorted partially and sorting is rare.

Table 2

Computation time for particles motion in 2D space without sorting and for sorting every 20 steps

Time without sorting, ms	62
Time for sorted particles, ms	10

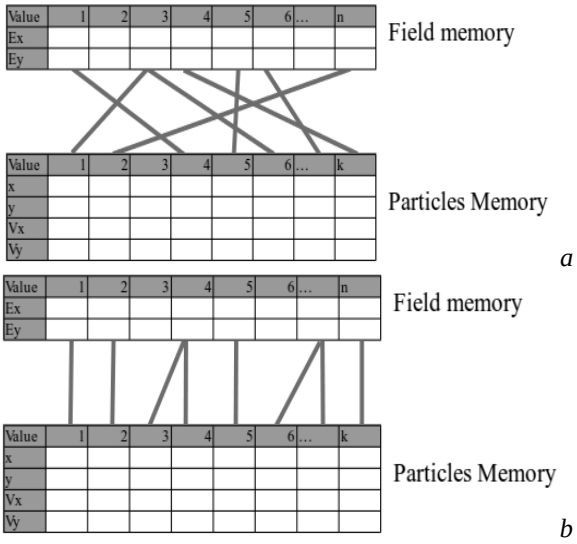

Fig 1. The memory accesses in the typical PIC code (a) and after 2D particles sorting (b)

2. GENERAL-PURPOSE COMPUTING ON GRAPHICS PROCESSING UNITS

GPGPU is very efficient for PIC methods because:

1. 2D and 3D vector operations are the built-in functions on GPU and can be performed by one step.
2. Linear and bilinear interpolations used in PIC for charges' and forces' weighting are the built-in functions on GPU and can be performed by one step.
3. For the particle motion we use a multitude of very simple calculations. GPU is optimized for solving such problems. A huge number of simple processor kernels that are used in Single instruction, multiple data (SIMD) architecture can substantially improve the performance [7].

Some existing packages (PSC, PIConGPU) use GPGPU. The computation time for particles motion and linear weighting problem is given in Table 3.

Table 3

CPU and GPU computation time for different number of particles in 2D space

Particles' number	10^7	10^6
CPU time, ms	430	56
GPU time, ms	20	3.5

3. LOAD BALANCING AND PARALLEL PIC METHOD

When we use parallelization, the problem of the system balancing appears. Let us consider the case of significantly inhomogeneous plasma (Fig. 2,a). If the modeling area will be decomposed into domains of the same square, the amount of data in each computing node will differ substantially. In this case the run-time calculations on each node will be different. Nodes performing calculations faster will be idle while waiting for the other nodes. So we need to balance the system by dividing the area into domains with roughly equal number of particles (Fig. 2,b). This problem was considered only in PSC package [3,4].

However, there is a problem that was not considered in any of the packages. Different Debye radii can occur for different domains. In this case, one should use meshes of different steps for optimal usage of the computing resources (Fig. 3). We developed an algorithm based on [8], allowing to solve the Poisson equation in the case of grids with different steps.

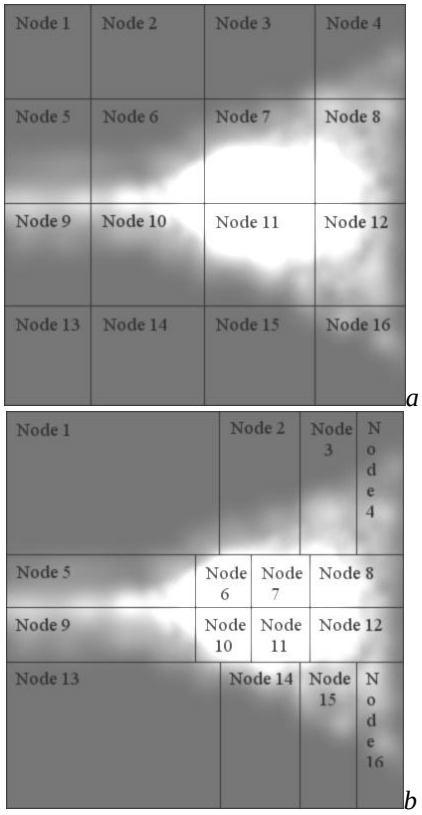

Fig 2. Unbalanced (a) and balanced (b) domain decomposition (background brightness corresponds to the plasma density)

4. FIELD EQUATION SOLVER

The significant feature of the most advanced programs is the use of the finite-difference time-domain method (FDTD). This approach is efficacious and simple for parallelization. However, many problems can't be considered by this method. For example, modeling of the electrostatic problem can't be performed by the FDTD methods because the divergence equation of electrostatic field can't be solved via this method. This is a significant disadvantage of PSC, PICCANTE and PIConGPU packages.

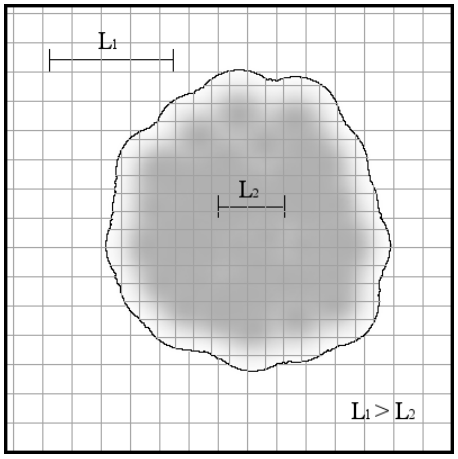

Fig 3. System with different Debye length

Another common approach to the parallel solving of the Poisson equation uses iterative methods such as multigrid, conjugate gradients, or alternating direction implicit methods. These methods can be efficiently parallelized. However, iterative methods' productivity is very low and does not allow to use them for the large size problems. This is a significant disadvantage of XOOPIC and modern power CPIC packages.

Table 4

Computation time comparison for common Poisson equation solvers (1024×1024 grid). Abbreviations: CR – cyclic reduction, ICR – incomplete cyclic reduction, MG – multi grid, CG – conjugate gradient

Method	Complexity	Time, s
CR	$O(N^2 log_2 N)$	0.18
Fourier	$O(N^2 log_2 N)$	0.2
ICR	$O(N^2 log_2 N)$	0.07
MG	$O(N^2 \ln e^{-1})$, e is the rate of convergence.	308
CG	$O(N^3)$	215
Proposed	$O(N^2 log_2 N)$	0.24

So, efficient parallel solution for electrostatic problem is not represented at the moment in any existing package. A direct method for Poisson equation solutions based on the domain decomposition and joining [9] was proposed [10]. The comparison of full (successive) computation time for proposed method and other common methods is presented in Table 4.

CONCLUSIONS

1. Cache misses avoiding is very important problem for PIC methods which can improve the performance up to 10 times. This problem wasn't solved in the existing simulation packages. The method for reducing the number of misses via periodical sorting is proposed. The Peano curve mapping 2D space to 1D one is used.

2. GPGPU using can improve the performance more than 30 times. There are packages containing GPGPU technique, but their application is limited by methods used for solving of the field equations.

3. Balancing problem should be solved to obtain the high performance load. This problem is solved in PSC package, but it uses FDTD field solving and can't simulate the electrostatic problems.

4. The problem of parallelization for field equation solving is still actual. Good parallelization was obtained only for solving of the wave propagation in plasma by FDTD method. The efficient parallel algorithm for solving Poisson equation was not yet proposed in the existing packages. We have developed an algorithm based on domain decomposition [9] for Poisson equation solving with high performance [10].

5. There is no package including all techniques (GPGPU or cache misses avoiding for CPU, load balancing, Poisson equation parallelization) for maximum performance. The new package should be developed that will have substantially higher performance.

REFERENCES

1. P.J. Mardahl, J.P. Verboncoeur. Progress in Parallelizing XOOPIC // *EECS Department, University of California*, Berkeley, CA 94720-1772, USA.

2. A. Sgattoni, L. Fedelia, et al. PICCANTE – an Open Source Particle-in-Cell Code for Advanced Simulations on Tier-0 Systems // *Technical report, PRACE white papers.* Accessed 23 May 2015. arXiv:1503.02464 [cs.DC] (online).

3. K. Germaschewski, W. Fox, S. Abbott, N.Ahmadi, K. Maynard, L. Wanga, H. Ruhl, A. Bhattacharjee. The Plasma Simulation Code: A modern particle-in-cell code with load-balancing and GPU support // *arXiv preprint,* 12 Nov 2015. arXiv:1310.7866 [physics.plasm-ph] (online).

4. S.J. Plimpton, D.B. Seide, et al. A load-balancing algorithm for a parallel electromagnetic particle-in-cell code // *Computer Physics Communications*. 2003, v. 152, p. 227-241.

5. PIConGPU: A Fully Relativistic Particle-in-Cell Code for a GPU Cluster // *IEEE Transactions on Plasma Science.* 2010, v. 38, issue 10, p. 2831-2839. doi: 10.1109/TPS.2010.2064310.

6. G.L. Delzanno, E. Camporeale, J.D. Moulton, et al. CPIC: A Curvilinear Particle-in-Cell Code for Plasma-Meterial Interaction Studies // *IEEE Transactions on Plasma Science.* 2013, v. 41, № 12.

7. D.A. Patterson, J.L. Hennessy. *Computer Organization and Design, Fourth Edition: The Hardware/Software Interface*. Morgan Kaufmann. 2011, p. 57-475.

8. Hertel, Stefan. Smoothsort's behavior on presorted sequence // *Information Processing Letters.* 1983, v. 16 (4), p. 165-170. doi:10.1016/0020-0190(83)90116-3.

9. N.V. Snytnikov. A parallel algorithm for solving 2D Poisson's equation in the context of no stationary problems // *Computational Methods and Programming.* 2015, v. 16, p. 39-51.

10. D.I. Dadyka, I.O. Anisimov. Direct parallel Poisson solver with the multiply grid spacing support for plasma simulation via PIC method // *International Conference – School on Plasma Physics and Controlled Fusion. Book of Abstracts.* Kharkov, Ukraine. September 12-15, 2016, p. 72.

Article received 05.10.2016

ОСОБЕННОСТИ РЕАЛИЗАЦИИ СОВРЕМЕННЫХ ПАКЕТОВ МОДЕЛИРОВАНИЯ ПЛАЗМЫ МЕТОДОМ КРУПНЫХ ЧАСТИЦ В ЯЧЕЙКАХ ДЛЯ ПАРАЛЛЕЛЬНЫХ СИСТЕМ

Д.И. Дадыка, И.А. Анисимов

Проведено сравнение распространённых программ с открытым исходным кодом для моделирования плазмы методом частиц в ячейках на параллельных вычислительных системах. В частности, рассмотрены вопросы решения уравнений поля, динамической балансировки и вычислений на графических ускорителях. Отмечены недостатки существующих программ: отсутствие эффективных методов решения электростатической задачи (уравнения Пуассона), неоптимальное использование кеширования, проблема моделирования существенно неоднородной плазмы. Рассмотрено влияние промахов кэша на производительность PIC-кодов. Предложен способ минимизации промахов путём периодической сортировки частиц в двухмерном пространстве. Предложен подход к эффективному параллельному решению уравнения Пуассона.

ОСОБЛИВОСТІ РЕАЛІЗАЦІЇ СУЧАСНИХ ПАКЕТІВ МОДЕЛЮВАННЯ ПЛАЗМИ МЕТОДОМ КРУПНИХ ЧАСТИНОК У КОМІРКАХ ДЛЯ ПАРАЛЕЛЬНИХ ОБЧИСЛЮВАЛЬНИХ СИСТЕМ

Д.І. Дадика, І.О. Анісімов

Виконано порівняння поширених програм із відкритим вихідним кодом для моделювання плазми методом частинок у комірках на паралельних обчислювальних системах. Розглянуто питання розв’язку рівнянь поля, динамічного балансування та обчислень на графічних прискорювачах. Відзначено недоліки існуючих програм: відсутність ефективних методів рішення електростатичної задачі (рівняння Пуассона), неоптимальне використання кешування, проблема моделювання суттєво неоднорідної плазми. Розглянуто вплив промахів кеша на продуктивність PIC-кодів. Запропоновано спосіб мінімізації промахів шляхом періодичного сортування частинок у двовимірному просторі. Запропоновано підхід до ефективного паралельного розв’язання рівняння Пуассона.