Parallelization of Gridless Finite-size-particle Plasma Simulation Codes

S. Briguglio*, G. Vlad, G. Fogaccia and B. Di Martino**

Associazione Euratom-ENEA sulla Fusione, C.R. Frascati,
C.P. 65 - I-00044 - Frascati, Rome, Italy.

Abstract. The main features of gridless finite-size-particle (FSP) codes are discussed, from the point of view of the performances that can be obtained, with respect both to the spatial-resolution level and the efficiency of parallel particle simulations. It is shown that such codes are particularly suited for particle-decomposition parallelization on distributed-memory architectures, as they present a strong reduction, in comparison with particle-in-cell (PIC) codes, of the memory and computational offsets related to storing and updating the replicated fluctuating-field arrays.

1 PIC and FSP Particle Simulation

Particle simulation codes [1] seem to be the most suited tool for the investigation of turbulent plasma behaviour. Particle simulation indeed consists in evolving, according to the equations of motion, the phase-space coordinates of a particle population in the fluctuating electromagnetic fields, selfconsistently computed, at each time step, in terms of the contribution yielded by the particles themselves (e.g., pressure perturbation), and allows one to fully retain all the relevant kinetic effects.

The equations for the fields can, in principle, be solved directly in the real space, using, e.g., finite difference methods. In many concrete situations, however, periodic spatial domains are considered. Moreover, in order to get a deeper insight in the physical mechanisms underlying the observed phenomena, the analysis is often focused on the evolution of relatively few harmonics. In such cases, it is worth solving the field equations in the Fourier space, then transforming the fields back to the real space in order to push the particles. In the present paper we want to show that, under this *few-harmonic* condition – which we assume to hold in the following –, a particular class of particle simulation codes, the *gridless finite-size-particle* (FSP) codes, is particularly suited to be efficiently parallelized on distributed-memory architectures.

The most widely used class of particle simulation codes is that of the *particle-in-cell* (PIC) codes, which compute the fluctuating fields and the required particle contribution only at the nodes of a spatial grid, then interpolating the fields

* briguglio@frascati.enea.it

** University of Naples "Federico II", Dipartimento di Informatica e Sistemistica, Naples, Italy

at the (continuous) particle positions in order to perform particle pushing. The introduction of a spatial grid yields a major benefit, in smoothing the short-range interactions between particles. In the absence of such a smoothing, these interactions would dominate over the long-range ones, due to the fact that, once ensured the coincidence of simulation and physical scale lengths by identifying each simulation particle with a macroparticle (i.e., a cluster of a large number of mutually non-interacting physical particles) that moves like a single physical particle [2], the density of the numerical plasma, n_0, is too low to satisfy the plasma condition $n_0 \lambda_D^3 \gg 1$ (with λ_D being the Debye length). This would not correspond to what happens in a physical plasma. However, it can be easily shown that, if L_c is the grid-point spacing, the condition that has to be satisfied in order to reproduce correct plasma behaviour is given by $n_0 L_c^3 \gg 1$, which replaces and relaxes the too stringent plasma condition. In an average sense, such a new condition can be written as $N_{part}/N_{cell} \gg 1$, where N_{part} and N_{cell} are the number of simulation particles and grid cells, respectively; it then corresponds to the requirement of disposing, for a given spatial resolution, of a number of particles large enough to accurately sample the velocity space. This requirement makes the full exploiting of parallel computers unavoidable as soon as high spatial-resolution levels have to be reached in order to investigate small-scale field fluctuations.

On the basis of these considerations, it is apparent that the main purpose of the parallelization of PIC codes is that of distributing the computational requests related to the particle population among several different processors. The computational effort related to the Fourier-space field solver is instead generally negligible and can be replicated on each processor; if this were not the case, also this portion of the simulation could be, in principle, distributed.

Two main different techniques have been developed in parallelizing PIC codes. The first approach is based on the *domain decomposition* [3]: different portions of the physical domain are assigned to different processors, together with the particles that reside on them. In this way, both the number of operations per time step and the memory space required to each computational node are reduced (in an average sense) by a factor equal, roughly speaking, to the number of processors, n_{proc}. Neglecting corrections due to inter-processor communication, these two quantities are indeed given by the following expressions:

$$N_{op\ d.d.}^{PIC} \approx f\left(N_{harm}\right) + \frac{1}{n_{proc}} \left(n_{FT} \times N_{harm} \times N_{cell} + n_{interp} \times N_{part}\right), \quad (1)$$

and

$$M_{d.d.}^{PIC} \approx m_{harm} \times N_{harm} + \frac{1}{n_{proc}} \left(m_{cell} \times N_{cell} + m_{part} \times N_{part}\right). \quad (2)$$

Here N_{harm} is the number of Fourier harmonics retained in the simulation; $f\left(N_{harm}\right)$ is an increasing function of N_{harm}, which corresponds to the number of operations required to update the electromagnetic fields in the Fourier space and whose precise dependence is determined by the structure of the problem and

the algorithm used by the field solver; n_{FT} is the number of operations needed to compute each addendum in the sum required by the Fourier transform (and anti-transform) [1] and n_{interp} that of the operations required for the interpolation of the fields (and for the distribution of the particle contribution among the grid points). Moreover, m_{harm}, m_{cell} and m_{part} are the amounts of memory needed to store, respectively, a single harmonic of the complete set of Fourier-space fields, the real-space fields at each grid point and the phase-space coordinates for each particle.

The main limitation of such a method consists in the fact that inter-processor communications are required when a particle migrates from one processor to another, and this can cause the parallel version of the code to be difficult to implement and/or inefficient. Moreover, serious load-balancing problems can arise because of such migration. If such problems do not occur, however, the main advantage of the scheme emerges, corresponding to the almost linear scaling of the attainable physical-space resolution with the number of processors. The computational effort required by the replicated Fourier-space solution of the field equations – corresponding to the first terms in Eqs. (1) and (2) – is indeed negligible in comparison with the one required by the grid and particle calculations – the terms multiplied by $1/n_{proc}$.

A complementary approach to the domain decomposition is represented by the *particle decomposition* [4]. It corresponds to replicate the whole spatial domain on each processor, while distributing the particle population. No particle migrates from one processor to another, because no particle meets, in its motion, the unphysical boundaries introduced by domain decomposition, and the communication among processors is both reduced and simplified (a little amount of communication is still required, because different-processor particle contributions to pressure have to be collected and summed together, at each time step, before updating fields). Moreover, load balancing is perfectly ensured during every simulation. On the opposite side the memory request associated to the grid quantities (equilibrium and fluctuating fields), which are replicated on each processor, gives rise to a bottle-neck on the scalability of physical resolution with processors. Neglecting the corrections due to the need for communicating partial pressure data, the computational load on each processor is indeed given, in this case, by

$$N^{PIC}_{op\ p.d.} \approx f\left(N_{harm}\right) + n_{FT} \times N_{harm} \times N_{cell} + n_{interp} \times \frac{N_{part}}{n_{proc}}, \qquad (3)$$

and

$$M^{PIC}_{p.d.} \approx m_{harm} \times N_{harm} + m_{cell} \times N_{cell} + m_{part} \times \frac{N_{part}}{n_{proc}}. \qquad (4)$$

Even in the limit of infinite number of processors, in which each processor must treat a vanishingly small number of particles, the maximum resolution that can

[1] We refer to general situations in which the Fast Fourier Transform (FFT) algorithm is not recommended (e.g., because the Fourier space is not densely filled). Our conclusions, however, would maintain their qualitative validity even in the FFT framework.

be reached is determined by the largest size of the (replicated) grid arrays that can be stored on each node.

Although the *few-harmonic* condition does not cause the real-space resolution request to be relaxed (one is anyway interested in the high mode-number portion of the complete fluctuation spectrum), it offers a different and more convenient path toward the target of an efficient parallelization and, hence, toward the high-resolution simulations. Instead of introducing a spatial grid for the calculations of the electromagnetic fields, and interpolating such fields at each particle position to perform particle pushing, it is possible to compute the actual values of the fields at the particle position by directly transforming them from the Fourier space back to the real space. Correspondingly, particle contributions to the pressure are just Fourier-transformed and summed together, instead of being collected at the nearest grid points. In such way, the memory resources demanded by the field storage are greatly reduced, and the bottle-neck in the scalability of the resolution with n_{proc} is in fact removed.

The suppression of the spatial grid, however, also eliminates the benefic smoothing of the short-range interactions between particles. To overcome such a difficulty, it is possible to resort to the FSP method [5], which consists in replacing the "singular" particles by "regular" *charge clouds* of typical size L_s. It is immediate to show that such a FSP ensemble behaves as a plasma under the condition $n_0 L_s^3 \gg 1$. PIC and FSP methods are expected to yield the same qualitative results if $L_c \approx L_s$.

A particle-decomposition approach to the parallelization of a gridless FSP code would then produce the following computational loads on each processor:

$$N_{op\ p.d.}^{FSP} \approx f\left(N_{harm}\right) + n_{FT} \times N_{harm} \times \frac{N_{part}}{n_{proc}}, \tag{5}$$

and

$$M_{p.d.}^{FSP} \approx m_{harm} \times N_{harm} + m_{part} \times \frac{N_{part}}{n_{proc}}. \tag{6}$$

In the case of serial simulations ($n_{proc} = 1$), considering that the quantity $f\left(N_{harm}\right)$ is typically negligible in comparison with the other terms that appear in Eqs. (3) and (5), and that for PIC codes $N_{part}/N_{cell} \gg 1$, it can be easily seen that, as far as $n_{FT} \times N_{harm} > n_{interp}$ (not too few harmonics), the gridless FSP method is more expensive than the PIC one, without offering any significant advantage in terms of lower memory requests. Both the relevant terms in Eq. (3) are indeed separately smaller than the term proportional to N_{part} in Eq. (5). This motivates the large diffusion of PIC codes and the poor consideration in which gridless FSP simulation has been taken until now.

In a parallel-simulation framework, however, the superiority of the FSP method is apparent. First of all, though FSP methods require back and forth Fourier transforms being executed for each particle, these calculations are distributed among different processors, differently from the replicated PIC grid calculations. From Eqs. (3) and (5), it can be seen that a gridless FSP code is

expected to become more advantageous than a PIC one for

$$n_{FT} \times N_{harm} \times N_{cell} + n_{interp} \times \frac{N_{part}}{n_{proc}} \gtrsim n_{FT} \times N_{harm} \times \frac{N_{part}}{n_{proc}}, \qquad (7)$$

or

$$n_{proc} \gtrsim N_{ppc} \left(1 - \frac{n_{interp}}{n_{FT} \times N_{harm}} \right), \qquad (8)$$

with N_{ppc} defined as $N_{ppc} \equiv N_{part}/N_{cell}$. In the FSP case, N_{cell} has, by itself, no meaning. However, we can interpret such number as the number of cells that would be necessary to accurately describe the same modes we investigate by the gridless code; in other words, while, in PIC simulations, $N_{cell} \propto L_c^{-3}$, in FSP simulations we can take $N_{cell} \propto L_s^{-3}$. We can then still formally refer to N_{ppc} as to an average number of particles "per cell", and look at N_{part} as the product of certain levels of spatial resolution (measured by N_{cell}) and velocity-space one (measured by N_{ppc}).

Second, different from the PIC case, large efficiency can be obtained even for massively parallel simulations, as far as the number of modes, N_{harm}, retained in the simulation is, in spite of the high mode numbers considered (high spatial resolution), relatively small; such a *few-harmonic* condition makes indeed the terms related to N_{harm} in Eqs. (5) and (6) negligible, if compared with the others and, in particular, causes the linear scaling of the maximum allowed spatial resolution with n_{proc} to be recovered even for a very large number of processors. At the same time, of course, the positive feature of an automatic load balancing, typical of the particle-decomposition parallelization, is preserved in this case.

2 The FSP Hybrid MHD-Gyrokinetic Code

In this section we present the main features of a parallel gridless FSP version of the Hybrid MHD-Gyrokinetic Code (HMGC), a code for the investigation of Alfvénic turbulence in magnetically confined plasmas, previously implemented in a PIC version [6]. Particle-decomposition parallelization of HMGC has been presented in Ref. [4]. The code solves the coupled sets of MHD equations for the fluctuating electromagnetic fields and gyrokinetic equations of motion for collision-free energetic ions. The physical domain is represented by a three-dimensional toroidal grid, for which quasi-cylindrical coordinates are adopted (see Fig. 1): the minor-radius coordinate, r, and the poloidal and toroidal angles, ϑ and φ, respectively. The field solver uses finite differences in the minor radius direction and Fourier expansion in the poloidal and toroidal directions.

The FSP version of the code eliminates the grid in the ϑ and φ directions, and introduces finite sizes for particles along the same directions. The corresponding Fourier variables are $k_\vartheta \equiv m/r$ and $k_\varphi \equiv n/R$, where m and n are, respectively, the poloidal and the toroidal number, and R is the major-radius coordinate of the torus. The radial grid is instead maintained (so that this FSP code is not

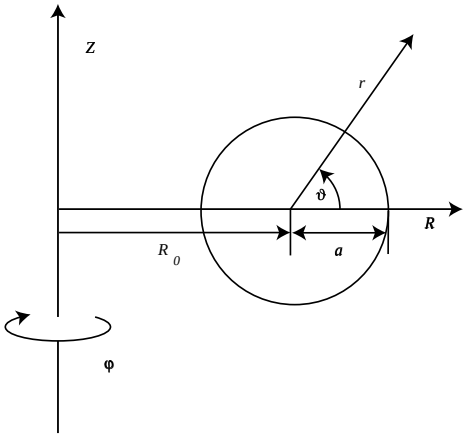

Fig. 1. Toroidal coordinate system (r, ϑ, φ) for a magnetically confined plasma with major radius R_0 and minor radius a.

a "gridless" one, in an absolute sense), because of the impossibility of adopting Fourier expansion in that direction.

The Alfvén modes, which constitute the main object of the investigation performed by HMGC, are typically characterized [7], for a given value of the toroidal number n, by the coupling of several poloidal harmonics, with sharper radial structure the higher the number n is, and poloidal numbers such that $nq(0) - 1/2 \leq m \leq nq(a) + 1/2$ (here $q(r) \equiv rB_\varphi / R_0 B_\vartheta$ is the so-called *safety factor*, a and R_0 are the minor and the major radius of the torus, respectively, B_φ is the toroidal magnetic field component and B_ϑ is the poloidal one). As a consequence, both poloidal and radial resolution requests (linearly) increase with increasing n, as it happens – of course – for toroidal resolution. In principle, memory resources of each processor could still be saturated by the offset associated to the first term in Eq. (6). Anyway, such an offset is in fact negligible, as its mode-number dependence, for a number of retained modes proportional to n, is quadratic (both m_{harm}, which takes into account the radial dependence of each mode, and N_{harm} scale linearly with n). In the PIC case, instead, the offset is dominated by the term $m_{cell} \times N_{cell}$, which scales as n^3. More precisely, with the level of precision adopted in our simulations to describe the spatial structure of the modes, the memory (in Megabytes) needed to store the field arrays for the FSP and the PIC version of HMGC, respectively, scales with the toroidal number n according to the following expressions:

$$M_{field}^{PIC} \approx 0.37 n^3, \tag{9}$$

and

$$M_{field}^{FSP} \approx 0.023 n^2. \tag{10}$$

Here we consider linear simulations with modes characterized by a single n, $q(0) = 1.1$ and $q(a) = 1.9$. It can be appreciated that the PIC approach is in fact penalized both by the n dependence and the proportionality coefficient. As such arrays are replicated on the different processors, the saturation of memory space (and the need for *memory paging*) is reached, in the PIC case, at lower values of n and, thus, at lower space resolution. The whole memory needed to store the distributed arrays associated to particle quantities (corresponding to the last term in Eqs. (4) and (6)) scales instead, for both versions of HMGC, according to the following expression:

$$M_{part} \approx 0.23 N_{ppc} \times n^3, \tag{11}$$

with the cubic dependence on n being associated to the "number of cells", $N_{cell} \equiv N_{part}/N_{ppc}$.

In Ref. [8] the PIC version of the code has been applied to the investigation of the linear stability and the nonlinear saturation of Alfvén modes in Tokamaks, in the presence of an energetic-ion population. The main results obtained in that analysis can be summarized as follows: as the energetic-ion pressure increases above a certain threshold, depending on the toroidal number n, the fast-growing Energetic Particle Mode (EPM) becomes unstable [7]. The EPM saturates because of a sudden displacement of a large part of the energetic particles along the minor radius. Such a displacement can, in principle, greatly enhance the energetic-particle losses, and makes the determination of the threshold for EPM destabilization an important issue in the theoretical research on reactor-relevant plasmas.

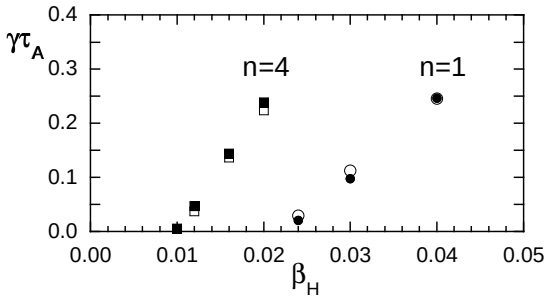

Fig. 2. Growth rate of Energetic Particle Modes, normalized to the inverse of the Alfvén time τ_A, obtained, at different values of n and β_H, by PIC (full symbols) and FSP (empty symbols) HMGC simulations.

Figure 2 shows the growth rate of the EPM's obtained by PIC and FSP HMGC simulations at different values of n and β_H (the ratio between the energetic-particle pressure and the magnetic one). The case of a plasma with aspect ratio $R_0/a = 10$ and energetic-particle Larmor radius ρ_H such that $\rho_H/a = 0.01$ is

considered here. A density profile of the form $\exp\left[-\left(r^2/L_n^2\right)^{\alpha_n}\right]$ has been assumed for the energetic ions, with $a^2/L_n^2 = 2$ and $\alpha_n = 2$. Slight quantitative differences between the results obtained by the two methods can be traced back to the different algorithms adopted, but a substantial agreement is found.

3 Parallelization Results

The FSP HMGC has been parallelized, according to the same strategy adopted for the corresponding PIC version of the code [4], within the High Performance Fortran [9] (HPF) framework, and tested on a 16-node IBM SP parallel system, by using the IBM *xlhpf* compiler [10] (an optimized native compiler for IBM SP systems). Some obvious differences exist between the two parallel codes, related, for example, to the fact that, in the FSP case, the Fourier components of the pressure are directly calculated as a sum over the particle population, rather than transforming the corresponding sum computed at the spatial grid points [2].

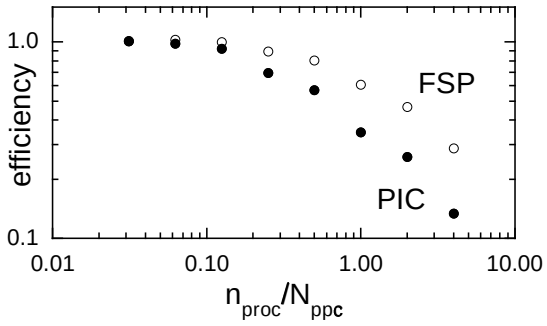

Fig. 3. Efficiency, defined as the speed-up factor divided by the number of processors, versus n_{proc}/N_{ppc}. Linear simulations retaining modes with a single toroidal number, $n = 1$, are considered. The results obtained by FSP HMGC (empty circles) are compared with those obtained by PIC HMGC (full circles).

The efficiency, defined as the speed-up factor divided by the number of processors, is plotted, in Fig. 3, versus n_{proc}/N_{ppc}, the inverse of the average number of particles per cell per processor. Linear simulations retaining modes with a single toroidal number, $n = 1$, are considered here. The results obtained by FSP HMGC are compared with those obtained by PIC HMGC. For such latter case, it was observed in Ref. [4] that the efficiency is almost at its ideal value as far as

[2] The auxiliary array, introduced in Ref. [4] with an extra dimension related to the number of processors, represents, in this case, the partial contribution (yielded by those particles that reside on a certain processor) to the Fourier transform of the pressure, rather than to the pressure itself.

$N_{ppc} \gg n_{proc}$, whereas it falls well below such a value when the average number of particles per cell seen by each processor is so small that the replicated grid calculations dominate over the distributed particle ones. It can be seen that, in the FSP simulations, the decrease in the efficiency is observed above a certain threshold in n_{proc}/N_{ppc}, significantly higher than the threshold found in the PIC case. This corresponds to the fact that the amount of replicated calculations is drastically reduced, and this fact allows to get high efficiency, at fixed number of particles, even at high number of processors. As a consequence, the particle-decomposition parallelization of FSP codes, different from the same parallelization of PIC codes, is suited even for massively parallel architectures.

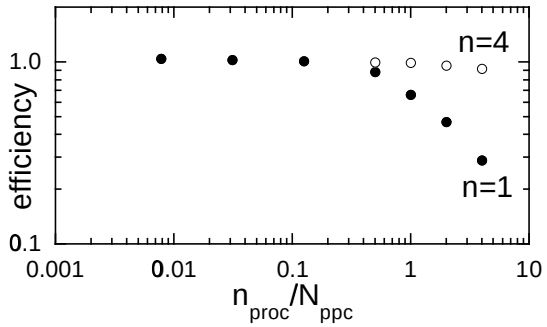

Fig. 4. Efficiency versus n_{proc}/N_{ppc} for FSP simulations with different values of the single toroidal number retained: $n = 1$ (full circles) and $n = 4$ (empty circles). The efficiency maintains almost its ideal values up to higher values of n_{proc}/N_{ppc} the higher the toroidal number n is.

Figure 4 plots the efficiency values, obtained in FSP simulations with different values of n, versus n_{proc}/N_{ppc}. Note that the efficiency threshold in n_{proc}/N_{ppc} comes out to increase with n. This fact can be easily understood, from Eq. (5), considering that the distributed part of the calculation scales with a higher power of n than the replicated one does. Therefore, the two parts become comparable (with a corresponding decrease of efficiency) for higher values of n_{proc}/N_{ppc} the higher the toroidal number n is.

Finally, Fig. 5 shows the Central Processor Unit (CPU) *user time* required by each simulation time step versus the number of processors. Results obtained by FSP (empty symbols) and PIC (full symbols) simulations with fixed mode number, $n = 1$, and different values of N_{ppc} (a), or fixed velocity-space resolution, $N_{ppc} = 4$, and different values of n (b) are compared. It can be seen that the FSP approach becomes more advantageous when the number of processors exceeds a certain threshold value. In qualitative agreement with Eq. (8) and with the fact that $N_{harm} \propto n$, such a threshold comes out to increase with N_{ppc} and, more weakly, with n.

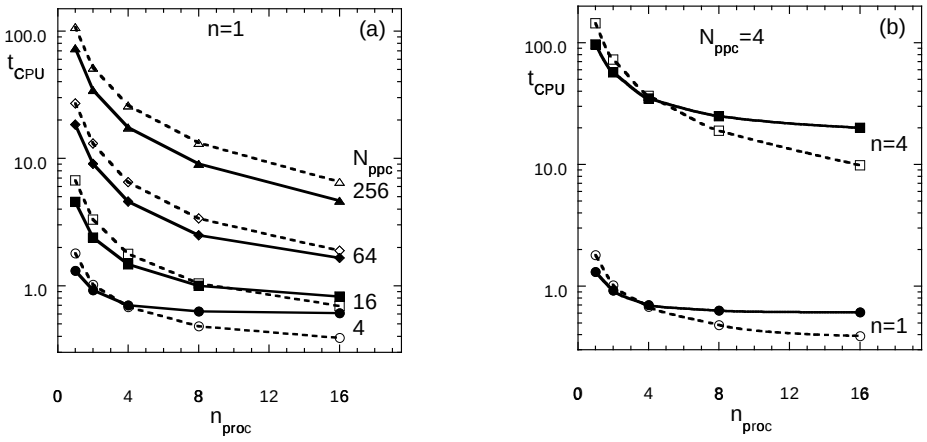

Fig. 5. CPU *user time* per time step versus n_{proc}. Results obtained by FSP (empty symbols) and PIC (full symbols) simulations with fixed mode number, $n = 1$, and different values of N_{ppc} (a), or fixed velocity-space resolution, $N_{ppc} = 4$, and different values of n (b) are compared.

References

1. C. K. Birdsall and A. B. Langdon, *Plasma Physics via Computer Simulation* (McGraw-Hill, New York, 1985).
2. A. Sestero, "Basic Interactions in Real Plasmas and in the Plasmas of Numerical Experiments", Il Nuovo Cimento 9B (1972) 222-232.
3. P. C. Liewer and V. K. Decyk, "A General Concurrent Algorithm for Plasma Particle-in-Cell Codes", J. Computational Phys. 85 (1989) 302-322.
4. B. Di Martino, S. Briguglio, G. Vlad and P. Sguazzero, "Parallel Plasma Simulation in High Performance Fortran", in *High Performance Computing and Networking*, (Springer, Berlin, 1998) 203-212.
5. A. B. Langdon and C. K. Birdsall, "Theory of plasma simulation using finite-size particles", Phys. of Fluids 13 (1970) 2115-2122.
6. S. Briguglio, G. Vlad, F. Zonca, and C. Kar, "Hybrid magnetohydrodynamic-gyrokinetic simulation of toroidal Alfvén modes", Phys. Plasmas 2 (1995) 3711-3723.
7. L. Chen and F. Zonca, "Theory of Shear Alfvén Waves in Toroidal Plasmas", Physica Scripta T60 (1995) 81-90.
8. S. Briguglio, F. Zonca, and G. Vlad, "Hybrid Magnetohydrodynamic-Particle Simulation of Linear and Nonlinear Evolution of Alfvén Modes in Tokamaks", Phys. Plasmas 5 (1998) 3287-3301.
9. H. Richardson, "High Performance Fortran: history, overview and current developments", Tech. Rep. TMC-261, Thinking Machines Corporation, 1996.
10. M. Gupta, S. Midkiff, E. Schonberg, V. Seshadri, D. Shields,K.Y. Wang, W. M. Ching, T. Ngo, "A HPF Compiler for the IBM SP2", in: Proc. SuperComputing '95 (ACM, 1995).