PARALLEL PARTICLE-IN-CELL (PIC) CODES*

F. Wolfheimer, E. Gjonaj, T. Weiland
Technische Universitaet Darmstadt, Institut fuer Theorie Elektromagnetischer Felder (TEMF)
Schlossgartenstr.8, 64289 Darmstadt, Germany

Abstract

A general methodology for the parallel performance modeling of PIC simulations is proposed. The performance model allows for the construction of an optimized parallelization approach for PIC which preserves an equal distribution of the computational workload on the processors while minimizing the interprocess communication. The general scheme is further specialized to simulations with a spatially very localized particle distribution as occuring in electron accelerators. The performance of the algorithm is studied using a benchmark problem. Additionally, a parallel, fully 3D simulation of the PITZ-Injector [1] is shown.

INTRODUCTION

Particle-In-Cell (PIC) simulations are commonly used in the field of computational accelerator physics for modeling the interaction of electromagnetic fields and charged particle beams in complex accelerator geometries. The modeling of charged particle dynamics requires the coupled solution of the MAXWELL equations

$$\begin{array}{rclcrcl} \nabla\times\vec{E} &=& -\frac{\partial}{\partial t}\vec{B}\,, & & \nabla\times\vec{H} &=& \vec{J}+\frac{\partial}{\partial t}\vec{D}, \\ \nabla\cdot\vec{D} &=& \rho\,, & & \nabla\cdot\vec{B} &=& 0 \end{array} \tag{1}$$

and the NEWTON-LORENTZ equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\vec{p}_i = q_i\left(\vec{E}(\vec{r}_i)+\frac{\vec{p}_i}{m_i}\times\vec{B}(\vec{r}_i)\right), \tag{2}$$

where $\vec{p}_i$, $\vec{r}_i$ describe the momentum and position and q_i, m_i express the charge and mass of particle i, respectively. The relation between the particle positions and momenta is given by $\vec{p}_i = m_i\frac{\mathrm{d}}{\mathrm{d}t}\vec{r}_i$. The coupling of equations (1) and (2) occurs through the source terms $\vec{J}$ and ρ. In the case of N_P particles the source terms may be expressed as

$$\vec{J} = \sum_{i=1}^{N_p} q_i\frac{\mathrm{d}\vec{r}_i}{\mathrm{d}t}\cdot\delta(\vec{r}-\vec{r}_i), \quad \rho = \sum_{i=1}^{N_p} q_i\cdot\delta(\vec{r}-\vec{r}_i). \tag{3}$$

The PIC algorithm approximates the solution of the system of partial differential equations (1,2) numerically. The solution strategy for the MAXWELL equations is based on a spatial discretization of the problem domain (computational grid). Field degrees of freedom (DOFs) are computed only at a finite number of points determined by the computational grid. Usually, for the discretization of the time dependence an explicit time integration scheme is applied. In this work the spatial discretization has been performed using the Finite Integration Technique (FIT) on structured, Cartesian grids [2, 3] and the leap-frog scheme has been used for the time integration.

For the numerical solution of the NEWTON-LORENTZ equation computational particles are introduced. Each computational particle represents a large number of real particles. The time integration of particle trajectories is performed using the well known Boris scheme and for the calculation of the source term $\vec{J}$ the charge conserving algorithm [4] is used. For a more detailed description of the PIC algorithm the reader is referred to [5].

However, the practicability of the method for real world simulations is often limited by the huge size of accelerator devices and by the large number of computational particles needed for obtaining accurate simulation results. Thus, the parallelization of the PIC codes becomes necessary to permit the solution of such problems in a reasonable time.

PARALLELIZATION OF PIC

The parallelization of PIC using the Single Program Multiple Data (SPMD) programing model allows for the execution of the resulting program on a cluster of compute nodes connected via a high speed network. Theoretically, this offers access to nearly unlimited computing power. However, the assignment of the computational workload to the available compute nodes is highly critical for simulation efficiency. Therefore, a detailed understanding of the computational workload generated by PIC is required to realize this assignment efficiently. Figure 1 shows the stages of a parallel PIC algorithm.

Two different parallelization strategies for PIC have been proposed in the literature. The first one is based on the geometrical decomposition of the computational domain (partitioning) among the compute nodes. Each node is responsible for performing computational tasks on the field DOFs and computational particles contained within the respective subdomain (partition) [6]. The advantage of this strategy follows from the fact that the gather and scatter operations do not require interprocess communication and, consequently, they introduce no communication overhead. However, as the particle distribution changes during the simulation the workload of the particle pusher will become imbalanced. This not only deteriorates the parallel performance of the algorithm but, additionally, may lead to the failure of the calculation if one of the compute nodes runs out of memory. This may happen when the particle distribution is spatially localized on a small region of

* This work has been partially supported by DESY Hamburg.

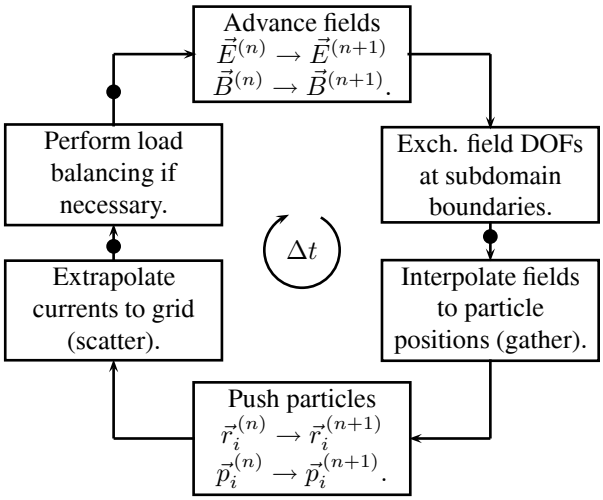

Figure 1: Flowchart for the parallelized PIC algorithm. Global synchronization points are marked with black dots.

the computational domain and, as a result, a single node has to hold all particles. Furthermore, particles leaving the subdomain associated to their owner processor have to be migrated to another processor and, therefore, the interprocess communication increases. A possibility to enhance the performance is to adaptively change the partition of the computational domain according to the particle distribution. While this may significantly improve the balance of the computational workload it likewise increases the interprocess communication because of the data exchange occuring during the load balancing operation. In some cases this additional costs even compensate the benefit of a better balanced workload. Further discussions concerning this approach may be found in [6].

The second strategy assigns the computational particles to processors independently from their position in the computational domain. The advantage of this strategy is the inherently balanced workload for both the field solver and the particle pusher. However, as the particles are assigned to arbitrary processors, the gather and scatter phases become nonlocal operations. A particle which is located in a cell for which its owner processor does not hold the field DOFs the required data has to be sent to by a remote processor. Thus, this approach provides a balanced workload while increasing the communication costs.

Other algorithms have been proposed which try to combine the strengths of both schemes. The reader is referred to [7] for a more detailed analysis of parallelization schemes for PIC.

Performance Modeling

A model describing the performance of a parallelization scheme can be helpful when comparing different possible implementation strategies and seeking for the one providing the best performance. In this section a performance model for PIC is introduced. The model is used to construct and optimize a parallelization scheme based on the uncoupled assignment of particles and field DOFs. Additionally, the model is used to predict the behaviour of the parallel speedup.

Symbol	Explanation
N_T	Total number of time steps.
$w_F^{(i_\pi)}(i_T)$	Time needed by processor i_π for field update during time step i_T.
$w_B^{(i_\pi)}(i_T)$	Time needed for the exchange of field DOFs assigned to cells located at the boundary of the local subdomain of processor i_π.
$w_{GS}^{(i\pi)}(i_T)$	Time needed for gather and scatter operations.
$w_P^{(i_\pi)}(i_T)$	Time needed for particle update.
$w_M^{(i_\pi)}(i_T)$	Time needed for the migration of particle and field DOFs due to, e.g., load balancing operations.
$n_C^{(i_\pi)}(i_T)$	Number of field DOFs assigned to processor i_π.
$n_B^{(i_\pi)}(i_T)$	Number of field DOFs which have to be exchanged for the field solver.
$n_P^{(i_\pi)}(i_T)$	Total number of particles pushed by processor i_π.
$\tilde{n}_P^{(i_\pi,j_\pi)}(i_T)$	Number of particles pushed by processor i_π which need field DOFs from processor j_π.
$m_C^{(i_\pi)}(i_T)$	Number of field DOFs migrated from and to processor i_π.
$m_P^{(i_\pi)}(i_T)$	Number of particles migrated from and to processor i_π.

Table 1: Explanation of the notations used for the performance model.

In the following performance model it is assumed that three global synchronization points per time step are introduced to the parallel PIC loop, cf. Fig. 1. The computing time between two synchronization points is determined by the processor which needs the most time to complete the tasks. The problem of workload assignment may be formulated as optimization problem. The objective function F which should be minimized is given by the total time needed to complete the simulation and can be written as

$$F := \sum_{i_T=1}^{N_T} \left(\max_{i_\pi} \left\{ w_F^{(i_\pi)}(i_T) + w_B^{(i_\pi)}(i_T) \right\} + \max_{i_\pi} \left\{ w_{GS}^{(i_\pi)}(i_T) + w_P^{(i_\pi)}(i_T) \right\} + \max_{i_\pi} \left\{ w_M^{(i_\pi)}(i_T) \right\} \right), \tag{4}$$

where the notation used in equation (4) is given in Tab. 1. The functions w_F, w_B, w_{GS}, w_P and w_M are assumed to

be linear in the number of particles and field DOFs, respectively, as follows:

$$w_F^{(i_\pi)}(i_T) = \alpha \cdot n_C^{(i_\pi)}(i_T) \tag{5}$$

$$w_B^{(i_\pi)}(i_T) = \eta \cdot n_B^{(i_\pi)}(i_T) \tag{6}$$

$$w_P^{(i_\pi)}(i_T) = \beta \cdot n_P^{(i_\pi)}(i_T) \tag{7}$$

$$w_M^{(i_\pi)}(i_T) = \delta \cdot m_C^{(i_\pi)}(i_T) + \epsilon \cdot m_P^{(i_\pi)}(i_T), \tag{8}$$

$$w_{GS}^{(i_\pi)}(i_T) = \sum_{\substack{j_\pi = 1 \\ j_\pi \neq i_\pi}}^{N_\pi} \left(\gamma \cdot \tilde{n}_P^{(i_\pi, j_\pi)}(i_T) + \gamma \cdot \tilde{n}_P^{(j_\pi, i_\pi)}(i_T) \right), \tag{9}$$

where $\alpha, \beta, \gamma, \delta, \epsilon$ and η are parameters determined by the hardware performance of the compute nodes and of the interconnection network. The fact that the model parameters do not depend on i_π corresponds to the (usual) case of a homogeneous cluster. Also, these parameters are normally time dependent as they may be influenced by other processes executed on the cluster. However, as it is impractical to model those frequent fluctuations, these parameters should be interpreted as mean values. The exchange of field data needed by the particle pusher results *vice versa* in the exchange of the currents created by the movement of the particles. Thus, the communication costs for the gather and scatter operations can be described by a single function. A migration of particles and field DOFs between processors, as described by w_M, appears during load balancing operations. The modeling of $w_F, w_B, w_P, w_{GS}, w_M$ as linear functions neglects any effects related to the memory hierarchy of modern computers (cache effects) as well as any latency of message passing operations. Tests performed on different clusters suggest that those assumptions are justified for PIC simulations of reasonable size.

The exact solution of the optimization problem formulated in equation (4) requires not only the knowledge of the hardware performance of the compute nodes and the interconnection network but also the *a priori* knowledge of the particle dynamics. As at least the information regarding the particle dynamics is unavailable prior to the simulation, the optimization problem can not be solved exactly. However, the problem can be solved when a few simplifications are introduced. For the following discussions only conditions where the field solver part is ideally balanced, i.e., the same number of field DOFs is assigned to each processor ($w_B^{(i_\pi)}(i_T) + w_F^{(i_\pi)}(i_T) = \text{const.}$), are considered. Furthermore, communication costs caused by the reassignment of particles are neglected ($w_M^{(i_\pi)}(i_T) = 0$). The objective function simplifies to

$$F = \sum_{i_T=1}^{N_T} \left(\max_{i_\pi} \left\{ w_{GS}^{(i_\pi)}(i_T) + w_P^{(i_\pi)}(i_T) \right\} \right). \tag{10}$$

The computational loads assigned to each processor and the communication costs for the gather and scatter phase can be described by a directed graph as shown in Fig. 2 for a simulation using three processors. As the graph illustrates

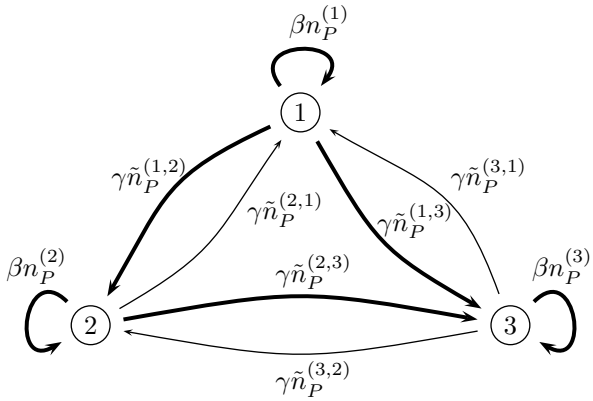

Figure 2: Graph illustrating the optimization problem.

the assignment of a particle to process j_π which needs field data from processor i_π where $i_\pi \neq j_\pi$ causes the computational load $\beta + \gamma$ for processor j_π but, additionally, it causes the computational load γ for processor i_π. Thus, for the case that $\gamma > \beta$, i.e., for a very slow interconnection network, it is not advantageous to assign any particle to a processor which does not own the field DOFs necessary for pushing the particle. In this case the parallelization scheme which couples the field DOFs and particle assignment with the geometrical partitioning of the domain outperforms the uncoupled scheme for an arbitrary particle distribution.

For the case that $\gamma < \beta$ a necessary condition for an optimum is that the load is balanced among the processors. Otherwise, the value of the objective function may be reduced by migrating particles from the most heavily loaded processor to the other processors. A necessary and sufficient condition for an optimal particle assignment in terms of the objective function requires the further constraint that the graph representing the workload does not possess any circles. Resolving a circle in the graph would result in a reduced workload for at least two processors and, thus, the value of the objective function may be reduced by the migration of particles from the other processors to the less loaded processors.

The following scheme which assigns the computational particles to the processors in N_π steps will result in an optimal state in terms of the objective function. The number of particles which depend on the field data possessed by processor i_π will be referred to as $r^{(i_\pi)}$. Without loss of generality it is assumed that the condition $r^{(i_\pi)} \leq r^{(j_\pi)}$ holds for all $i_\pi < j_\pi$. In an initial step $r^{(1)}$ particles are assigned to each processor such that only local field data is required for the pushing. After this step there are no more particles depending on field data from processor 1. In the next step a total of $(r^{(2)} - r^{(1)}) \cdot (N_\pi - 1)$ particles are assigned to the processors. To processors $2, .., N_\pi$ again only particles are assigned which depend on local field data. To processor 1 a number of particles requiring field data from processors $2, .., N_\pi$ is assigned where the number of particles requiring field data from processor i_π is equal to the number of particles requiring field data from processor j_π. In the

i-th step there are only particles left which depend on the field data of processors $i+1,..,N_\pi$. For processors $1,..,i$ the total number of particles assigned in this step which require field DOFs from one of the processors $i+1,..,N_\pi$ is given by

$$\Delta n_P^{(1..i)}(i) = \frac{\beta \cdot (N_\pi - i) \cdot (r^{(i+1)} - r^{(i)})}{N_\pi \cdot \beta + (N_\pi - 2i) \cdot \gamma}, \qquad (11)$$

while the number of particles assigned to processors $i,..,N_\pi$ is given by

$$\Delta n_P^{(i+1..N_\pi)}(i) = \frac{[\beta \cdot (N_\pi - i) + \gamma \cdot (N_\pi - 2i)] \cdot (r^{(i+1)} - r^{(i)})}{N_\pi \cdot \beta + (N_\pi - 2i) \cdot \gamma}. \qquad (12)$$

This assignment scheme results in an equally balanced workload and, additionally, introduces no circles into the graph. The edges present in the resulting graph are marked with bold lines in Fig. 2.

The following example illustrates the algorithm for a configuration with three processors ($N_\pi = 3$). The model parameters are assumed to have the values $\beta = 3$ and $\gamma = 1$. A total of 120 particles are to be assigned to the processors and the values for $r^{(i_\pi)}$ are $r^{(1)} = 20$, $r^{(2)} = 30$ and $r^{(3)} = 70$. In the initial step $N_\pi \cdot r^{(1)} = 60$ particles are assigned such that each processor obtains 20 particles depending only on local field DOFs. In the first step, the next $(r^{(2)} - r^{(1)}) \cdot (N_\pi - 1) = 20$ particles are considered. Following (11), processor 1 obtains $\frac{(N_\pi - 1)\cdot\beta}{\beta \cdot N_\pi + \gamma \cdot (N_\pi - 2)} \cdot (r^{(2)} - r(1)) = 6$ particles, i.e., 3 particles which require field DOFs from processor 2 and processor 3, respectively. The number of particles assigned to processors 2 and 3 is given by $\frac{\beta \cdot (N_\pi - 1) + \gamma \cdot (N_\pi - 2)}{\beta \cdot N_\pi + \gamma \cdot (N_\pi - 2)} \cdot (r^{(2)} - r^{(1)}) = 7$, as stated by (12). Those particles depend only on field DOFs local to processors 2 and 3. In the second step, processors 1 and 2 obtain 15 particles each which require field DOFs from processor 3. Consequently, processor 3 receives 10 particles requiring local field DOFs. Thus, the total load of the processors is

$$W^{(1)} = 41\beta + 21\gamma = 123 + 21 = 144, \qquad (13)$$
$$W^{(2)} = 42\beta + 18\gamma = 126 + 18 = 144, \qquad (14)$$
$$W^{(3)} = 37\beta + 33\gamma = 111 + 33 = 144. \qquad (15)$$

While the introduced scheme can be applied to general PIC simulations, for some classes of simulations the particle dynamics is approximately known prior to the simulation and, thus, this knowledge can be used to construct a scheme which is optimized for this class of simulations. In the following section a simplified version of the above scheme is introduced and analyzed.

Adaptive Bounding Box Approach

In the simulation of short bunch dynamics in accelerator devices, such as the PITZ-Injector [1], the particle distribution is often spatially very localized. For such simulations it can be advantageous to perform the necessary data exchange of the gather and scatter phase not particlewise as for the general case described in the previous section but rather by a global reduction operation. For that purpose, each processor computes a local bounding box which describes the part of the computational domain which is filled with the particles it owns during the push phase. This is a computationally very cheap operation. Following, a global bounding box is computed which describes the smallest rectangular domain containing the local bounding boxes of all processors. Figure 3 shows a small fraction of a computational domain. The global bounding box for the localized particle distribution has been marked by bold lines. Next,

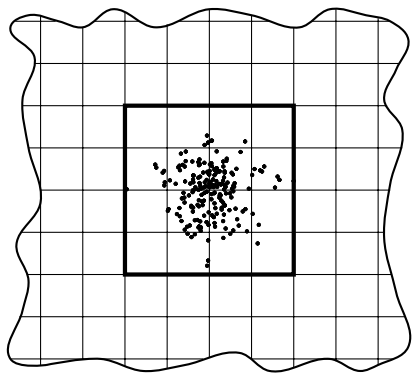

Figure 3: The global bounding box describes the smallest rectangular domain which contains all particles.

each processor which possesses field DOFs located inside the bounding box sends those values to all other processors via a global reduction operation provided by the underlying message passing library. Knowing the field DOFs inside the bounding box each processor is able to push the particles it owns. The currents excited by the particle movement are collected using again a global reduction operation. A big advantage of the approach is its simplicity as it does not require the organization of many complicated and error prone communication operations. Furthermore, the collective communication functions of message passing libraries are often implemented using efficient schemes such that the time needed to complete the operation scales less than linear with the number of processors. This is also true for the MPI [8] implementations used for the performance evaluation presented in the next section. Figure 4 shows the dependency of the completion time for the global reduction function used in the implementation from the number of processors and the length of the message, respectively. While the completion time depends linearly on the message length, the dependency from the number of processors is approximately $O(\log(N_\pi))$. Therefore, the term w_{GS} simplifies and can be expressed as

$$w_{GS} = \kappa \cdot N_{\text{bunch}} \cdot \log(N_\pi), \qquad (16)$$

where N_{bunch} describes the number of field DOFs included in the global bounding box and κ is a parameter describ-

ing the bandwidth of the interconnection network. Additionally, no data migration is required by the approach as the computational load remains balanced during the whole computation. Therefore, the term w_M in the objective function vanishes.

The very simple structure of the algorithm and the independence of the communication needed for the gather and scatter phases from both the assignment of field DOFs and particles to the processors allows for a prediction of the parallel speedup in case the overhead introduced by the exchange of field DOFs at subdomain boundaries is neglected ($w_B = 0$). The overhead associated with this term is hard to predict in a general expression as it depends completely on the assignment of the field DOFs to the processors. However, as the predicted speedup shows a good agreement with the performance results obtained with an implementation this assumption seems to be justified. With the functions in (5,7,16) and the simplifications described above the predicted parallel speedup $\hat{S}_{N_\pi}$ of the algorithm can be written as

$$\hat{S}_{N_\pi} = \frac{1}{\frac{1}{N_\pi} + \frac{\kappa \cdot N_{\text{bunch}}}{\alpha \cdot N_C + \beta \cdot N_P} \cdot \log(N_\pi)}. \qquad (17)$$

where N_C and N_P is the total number of field DOFs and particles used in the simulation, respectively. From (17) it becomes clear that the speedup of the algorithm is bound by the communication costs as the computational workload is ideally balanced. For an interconnection network with infinite bandwidth ($\kappa \to 0$) the algorithm scales optimally.

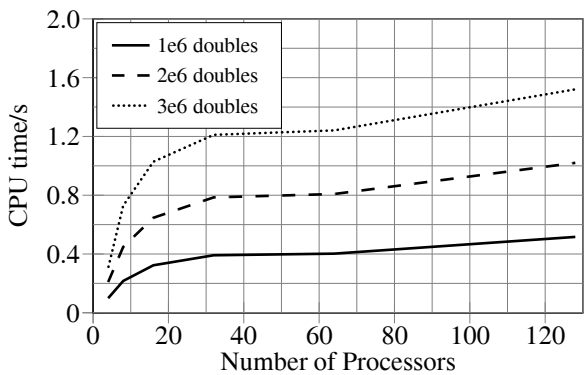

Figure 4: Performance of `MPI_Allreduce(...)`.

RESULTS

The performance of the proposed parallelization approach has been investigated using a benchmark example of an electron bunch traveling inside an ideally conducting tube. A model of the PITZ-Injector provides a real world simulation example. The simulations have been performed on the cluster of the Center for Scientific Computing (CSC) in Frankfurt/M.. This cluster is build up of 282 nodes with dual processor boards (AMD Opteron 244, 1.8 GHz), 64 nodes are connected via Myrinet and 218 nodes are connected via 1 Gbps Ethernet.

Performance Results

Figure 5 shows the dependency of the parallel speedup from the number of particles used in the simulation. As (17) suggests, the speedup is better on the faster Myrinet network. Besides, an increasing number of particles increases the speedup of the algorithm. Figure 6 shows the dependency of the speedup from the number of field DOFs included in the bounding box. The bounding box of the first bunch (size 1) includes about 600 field DOFs while the bounding box of the second bunch (size 2) includes about 4800 field DOFs. The spatial discretization using 5 million grid points remains unchanged for all simulations. The decrease of the parallel speedup when the bounding box increases can be obtained. The predicted behaviour of the parallel speedup given by equation (17) is in very good agreement with the results obtained by the benchmarks.

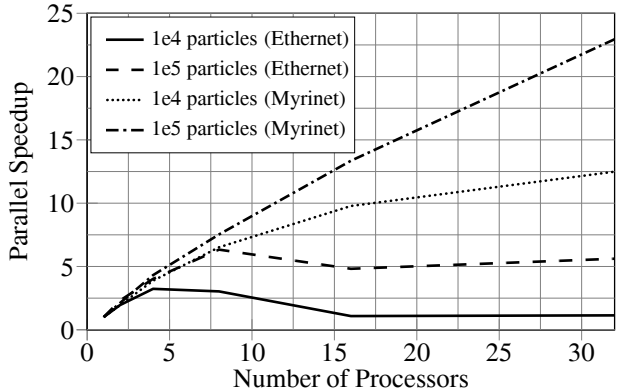

Figure 5: Influence of the bandwidth of the interconnection network on the parallel speedup of the bounding box approach.

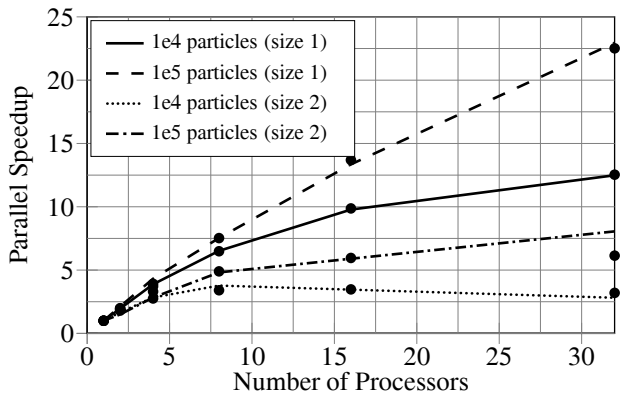

Figure 6: Influence of the number of field DOFs within the bounding box on the parallel speedup. The dots indicate the speedup values predicted by (17).

PITZ-Injector

PITZ is a test facility at DESY Zeuthen for research and development on laser driven electron sources for Free Electron Lasers (FEL) and linear colliders. Figure 7 shows a

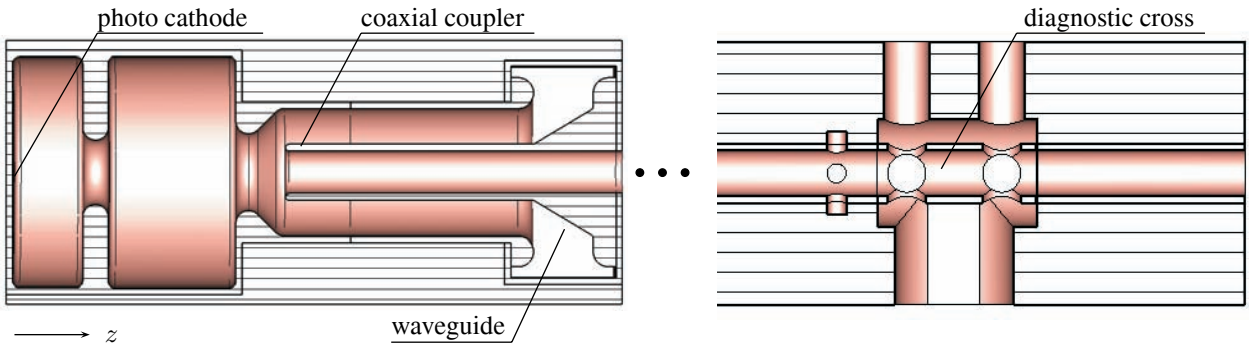

Figure 7: CAD model of the PITZ-Injector including the diagnostic cross.

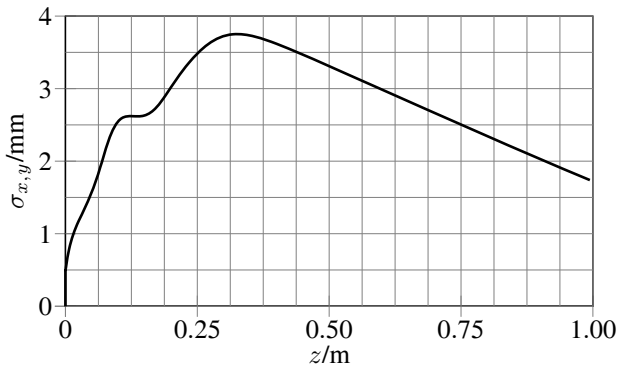

Figure 8: Transversal rms-radius of the electron bunch.

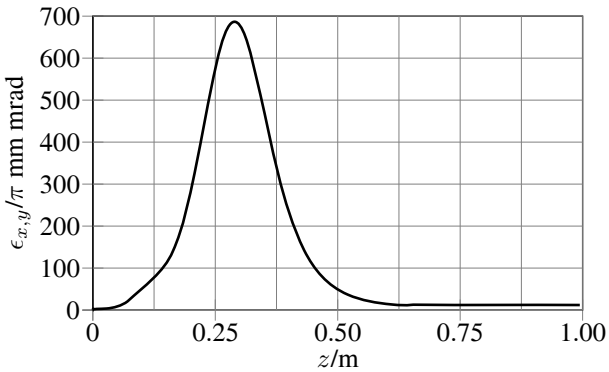

Figure 9: Transversal emittance of the electron bunch.

CAD model of the electron gun and the diagnostic cross used for the insertion of sensors into the beam line. The influence of the inhomogenities in the trift tube introduced by the diagnostic cross on the dynamics of the electron bunch is an important question. Thus, a full 3D PIC Simulation of the structure has been performed. The maximum dimensions of the electron bunch are $\sigma_{x,y} = 3.75$mm, $\sigma_z = 2.5$mm while the dimensions of the gun and the trift tube including the diagnostic cross are $l_{x,y} = 10$cm, $l_z = 1$m. Due to the multiscale nature of the problem an extremely high number of grid points is required to discretize the model with the necessary accuracy. As the particle distribution is spatially very localized, the parallelization approach described above is appropriate for this problem. Figure 8 and 9 show the transversal bunch rms-radius and the transversal emittance versus the z-position of the bunch barycenter.

CONCLUSIONS

A parallelization approach for PIC simulations has been presented and the performance has been evaluated in a computational environment of a computer cluster. Additionally, the theoretical analysis of parallel performance has been discussed. The analytically predicted parallel speedup has been found to be in good agreement with the performance results obtained on the computer cluster.

A real world simulation of the PITZ-Injector has been performed using the proposed parallel strategy.

REFERENCES

[1] A. Oppelt et al., "Status and First Results from the Upgraded PITZ Facility", Proc. FEL 2005

[2] T. Weiland, "A Discretization Method for the Solution of Maxwell's Equations for Six-Component Fields",Electronics and Communication, Vol.31, pp.116-121, 1977

[3] T. Weiland, "On the Numerical Solution of Maxwell's Equations and Applications in Accelerator Physics", Particle Accelerators, Vol.15, pp.245-292, 1984

[4] J. Villasenor and O. Buneman, "Rigorous charge conservation for local electromagnetic field solvers", Computer Physics Communications 69, pp.306-316, 1992

[5] U. Becker, T. Weiland, "Particle-in-Cell Simulations within the FI-Method", Surveys on Mathematics in Industry, Vol.8, No.3-4, pp.233-242, 1999

[6] F. Wolfheimer, E. Gjonaj, T. Weiland "Dynamic Domain Decompositions for Parallel PIC Simulations in the Time Domain", Proc. ICEAA, pp.1003-1006, 2005

[7] E. A. Carmona, L. J. Chandler, "On parallel PIC versatility and the structure of parallel PIC approaches", Concurrency: Practice and Experience, Vol.9(12), pp.1377-1405, 1997

[8] "MPI-2: Extensions to the Message-Passing Interface",University of Tennessee, Knoxville, Tennessee, 1997