Parallel Particle-In-Cell (PIC) Codes

Felix Wolfheimer, Erion Gjonaj and Thomas Weiland

International Computational
Accelerator Physics Conference (ICAP)

5th October 2006

Technische Universität Darmstadt, Fachbereich Elektrotechnik und Informationstechnik
Schloßgartenstr. 8, 64289 Darmstadt, Germany - URL: www.TEMF.de

Outline

- Motivation
- The Particle-In-Cell (PIC) Algorithm
- Parallelization and Performance Modeling for PIC
- Simulation and Performance Examples
- Conclusion

Outline

- Motivation
- The Particle-In-Cell (PIC) Algorithm
- Parallelization and Performance Modeling for PIC
- Simulation and Performance Examples
- Conclusion

Motivation

- It exists a continuous demand for

1. more computational resources to perform larger and more accurate simulations,
2. faster computations to perform more simulations in less time.

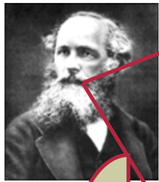
High Performance Computing provides a possibility to meet those requirements.

Outline

- Motivation
- The Particle-In-Cell (PIC) Algorithm
- Parallelization and Performance Modeling for PIC
- Simulation and Performance Examples
- Conclusion

TECHNISCHE UNIVERSITÄT DARMSTADT

Physical Model



Maxwell Equations

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

$\vec{E}, \vec{B}$

coupled nonlinear equation system

$\vec{J}, \rho$

Newton-Lorentz Equations

$$\begin{aligned}\frac{\mathrm{d}\vec{p}_i}{\mathrm{d}t} &= q_i\left(\vec{E}(\vec{r}_i) + \frac{\vec{p}_i}{m_i} \times \vec{B}(\vec{r}_i)\right)\\ \frac{\mathrm{d}\vec{r}_i}{\mathrm{d}t} &= \frac{\vec{p}_i}{m_i}\end{aligned}$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Particle-In-Cell Algorithm

Advance fields
$\vec{E}^{(n)} \rightarrow \vec{E}^{(n+1)}$
$\vec{B}^{(n)} \rightarrow \vec{B}^{(n+1)}$.

Interpolate fields to particles (gather).

Push particles
$\vec{r}_i^{(n)} \rightarrow \vec{r}_i^{(n+1)}$
$\vec{p}_i^{(n)} \rightarrow \vec{p}_i^{(n+1)}$.

Extrapolate currents to grid (scatter).

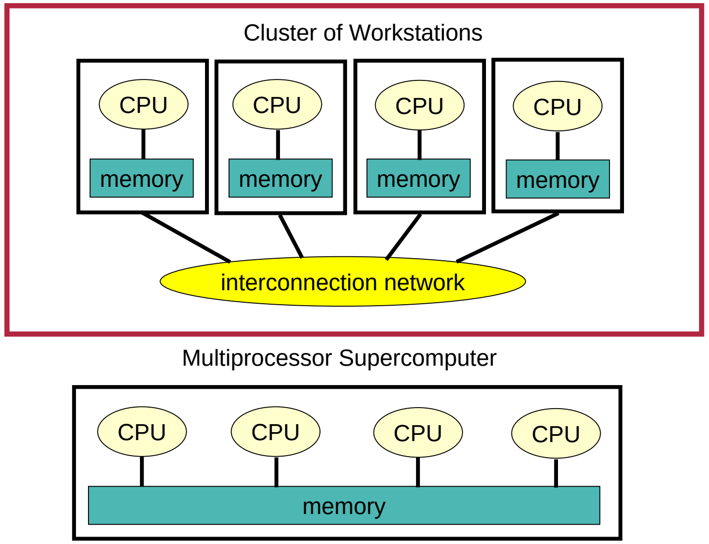
Δt

TECHNISCHE UNIVERSITÄT DARMSTADT

Outline

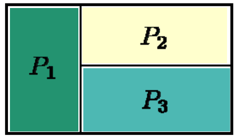
- Motivation
- The Particle-In-Cell (PIC) Algorithm
- Parallelization and Performance Modeling for PIC
- Simulation and Performance Examples
- Conclusion

Computational Environment



Parallelization Strategies

- The Parallelization of the field solver requires a geometrical partitioning of the computational domain.

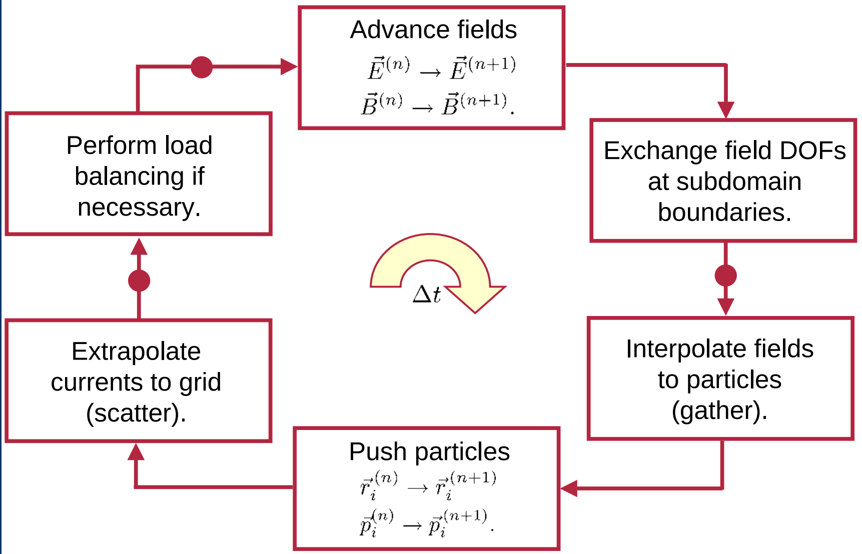


- Assignment of particles to processors is critical for simulation efficiency.

coupled assignment:
+ local gather/scatter phases
- imbalanced load

uncoupled assignment:
+ balanced load
- nonlocal gather/scatter phases

Particle-In-Cell Algorithm



Performance Modeling

- How should the particles be assigned to the available processors?

Objective Function

$$\text{minimize} \quad F := \sum_{i_T=1}^{N_T} \Bigg(\max_{i_\pi} \left\{ w_F^{(i_\pi)}(i_T) + w_B^{(i_\pi)}(i_T) \right\} + \max_{i_\pi} \left\{ w_{GS}^{(i_\pi)}(i_T) + w_P^{(i_\pi)}(i_T) \right\} + \max_{i_\pi} \left\{ w_M^{(i_\pi)}(i_T) \right\} \Bigg)$$

N_T - total number of timesteps

$w_{\dots}^{(i_\pi)}(i_T)$ - Time needed by processor i_π to complete phase ... in time step i_T .

Performance Modeling

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

- Assumptions:
  - Field solver is ideally balanced.
  - Migration costs are neglected.

Performance Modeling

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

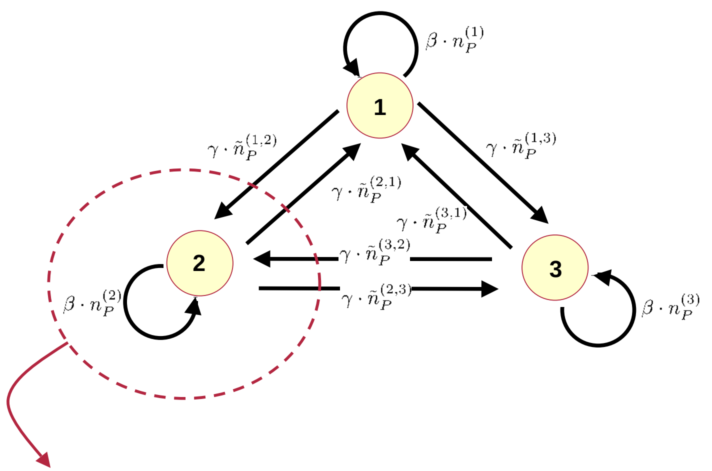
$$w_P^{(i_\pi)}(i_T) := \beta \cdot n_P^{(i_\pi)}(i_T)$$

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

- Assumptions:
  - Field solver is ideally balanced.
  - Migration costs are neglected.
  - For the particle pusher and the gather/scatter costs a linear dependency is assumed.
  - Computational environment is a homogeneous cluster.

Load Graph

- The workload can be illustrated by a directed graph.

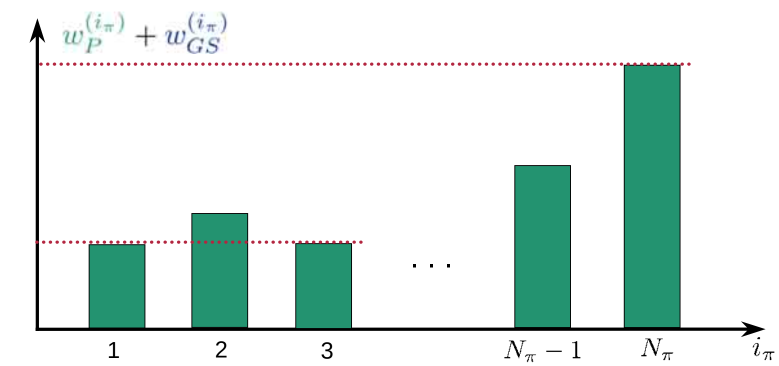


$$w_P^{(2)} + w_{GS}^{(2)} = \beta \cdot n_P^{(2)} + \gamma \cdot \left(\tilde{n}_P^{(1,2)} + \tilde{n}_P^{(2,1)} + \tilde{n}_P^{(2,3)} + \tilde{n}_P^{(3,2)} \right)$$

Optimization Procedure

First case $\gamma > \beta$ **:**

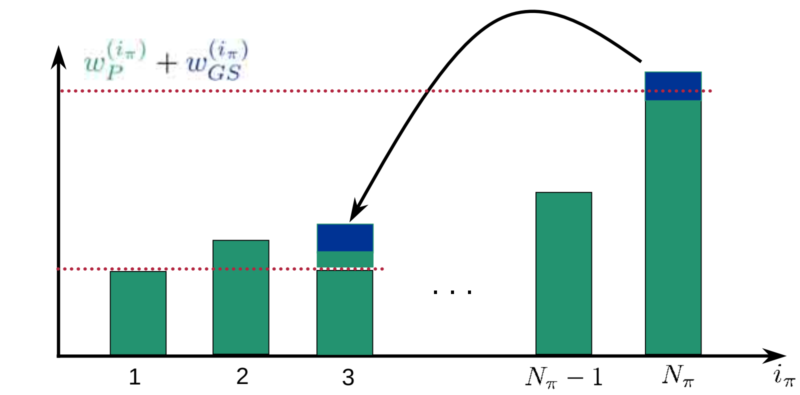
It is optimal to choose the coupled assignment and accept the unbalanced load for the particle pusher.



Optimization Procedure

First case $\gamma > \beta$ **:**

➡ It is optimal to choose the coupled assignment and accept the unbalanced load for the particle pusher.



TECHNISCHE UNIVERSITÄT DARMSTADT

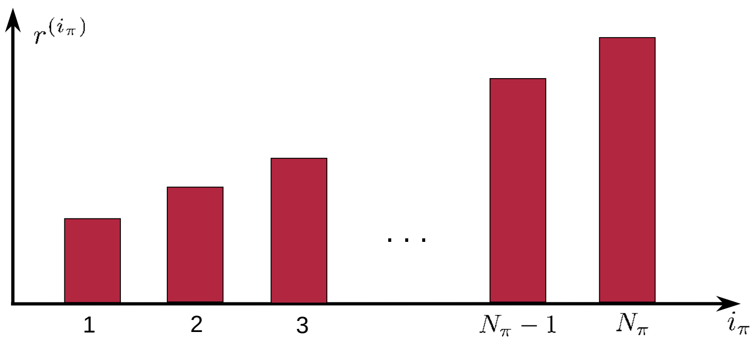
Optimization Procedure

Second case $\gamma \leq \beta$**:**

1. Sort processes such that

$$r^{(i_\pi)} \leq r^{(j_\pi)} \quad \forall i_\pi, j_\pi \in [1, .., N_\pi] : i_\pi < j_\pi,$$

where $r^{(i_\pi)}$ describes the number of particles depending on field DOFs from processor i_π .

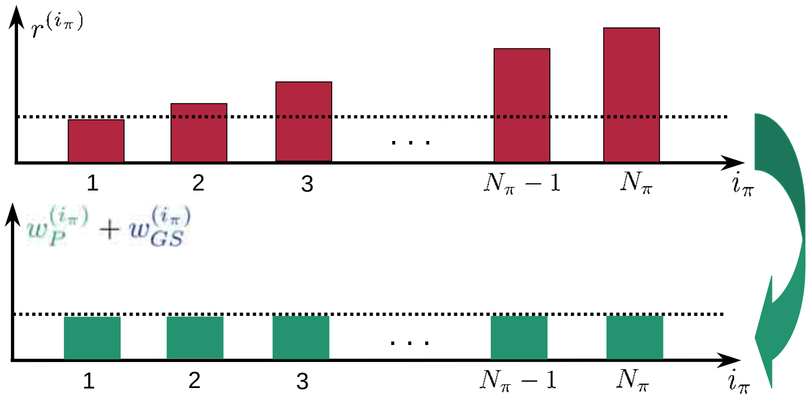


TECHNISCHE UNIVERSITÄT DARMSTADT

Optimization Procedure

Second case $\gamma \leq \beta$**:**

2. Assign the first $N_\pi \cdot r^{(1)}$ particles to the processors such that the particles depend only on local field DOFs.

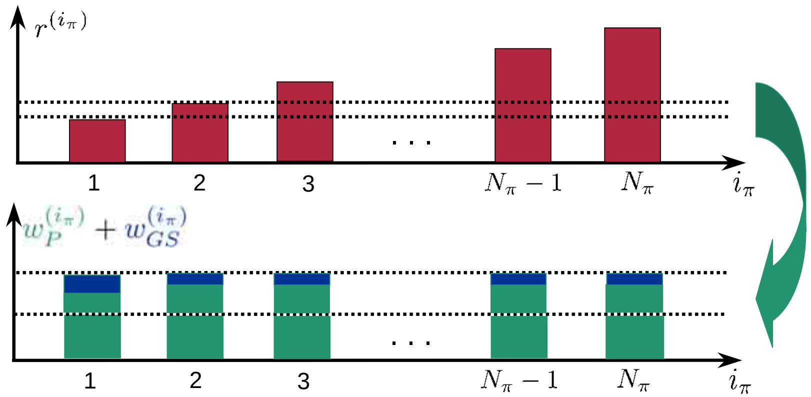


TECHNISCHE UNIVERSITÄT DARMSTADT

Optimization Procedure

Second case $\gamma \leq \beta$**:**

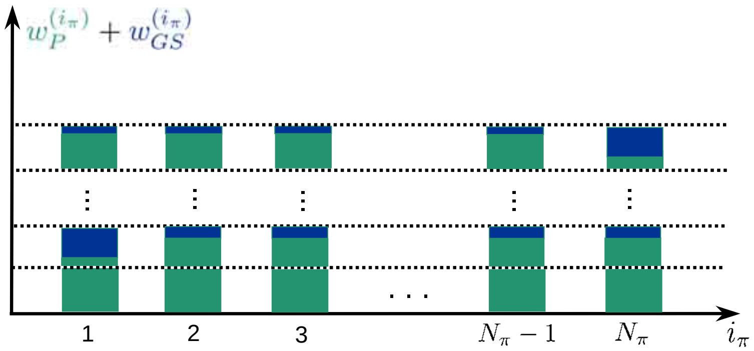
3. Assign $(N_\pi - 1) \cdot (r^{(2)} - r^{(1)})$ particles requiring field DOFs possessed by processors $2, .., N_\pi$. Preserve a balanced load.



Optimization Procedure

Second case $\gamma \leq \beta$:

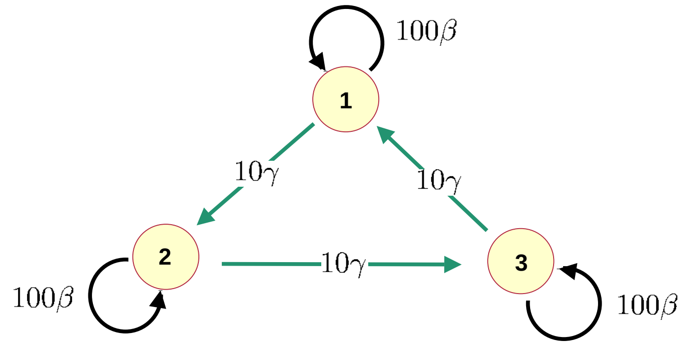
4. In the i^{th} step assign $(N_\pi - i)\cdot(r^{(i+1)} - r^{(i)})$ particles depending on field DOFs from processors $i+1, .., N_\pi$. Again, preserve a balanced load.



Optimization Procedure

- Is this the optimal solution in terms of the model?

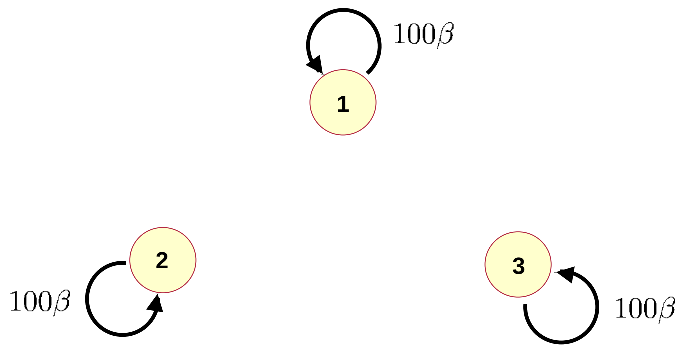
➡ Yes, it is. For all suboptimal strategies the Load Graph isn‘t load balanced or contains circles.



Optimization Procedure

- Is this the optimal solution in terms of the model?

➡ Yes, it is. For all suboptimal strategies the Load Graph isn‘t load balanced or contains circles.

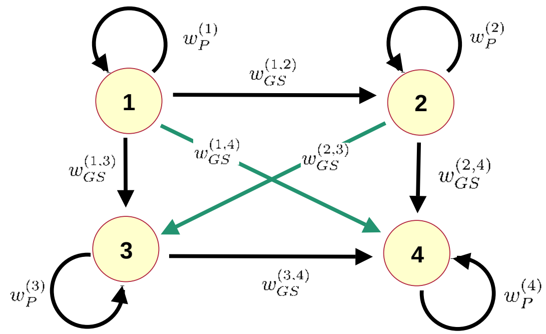


Optimization Procedure

- Is this the optimal solution in terms of the model?

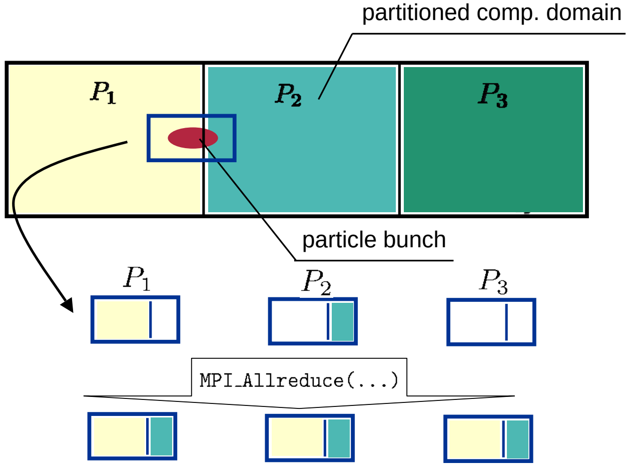
➡ Yes, it is. For all suboptimal strategies the Load Graph contains circles or isn‘t load balanced.

- But, the optimum is not unique for more than 2 processors.



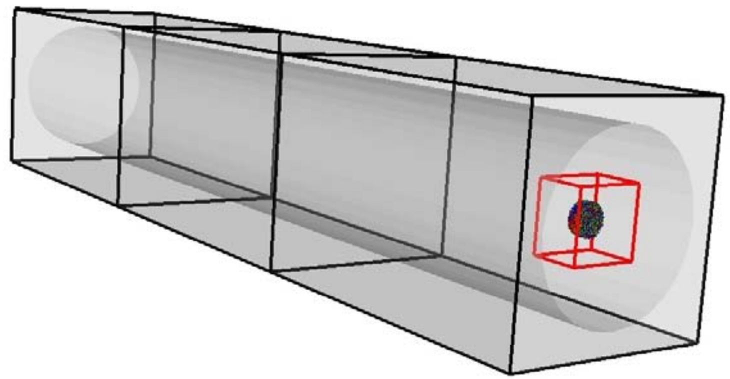
Adaptive Bounding Box

- Data exchange can be simplified for simulations with a very localized particle distribution.



Adaptive Bounding Box

- The Bounding Box is adaptively changed depending on the particle distribution.



Adaptive Bounding Box

- Communication costs can be approximated by

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

- The parallel speedup can be predicted by the expression

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

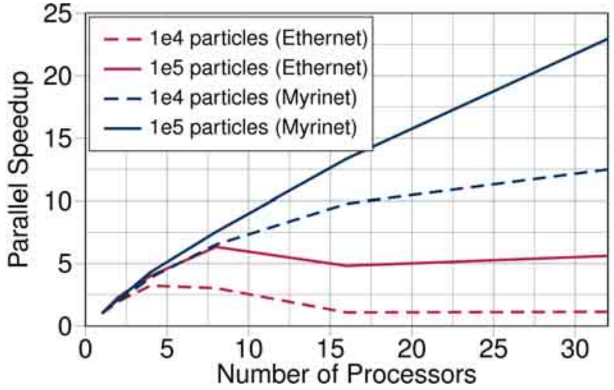
→ The algorithm is communication bound. Good performance results can only be expected for a fast interconnection network.

Outline

- Motivation
- The Particle-In-Cell (PIC) Algorithm
- Parallelization and Performance Modeling for PIC
- Simulation and Performance Examples
- Conclusion

Benchmark Results

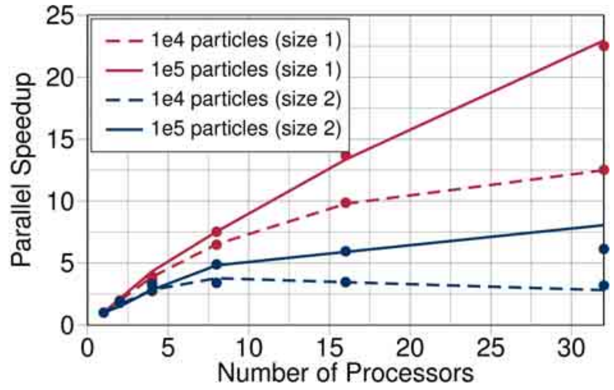
- Benchmark: electron bunch traveling within an ideally conducting tube (5 million grid points).



➡ The performance of the algorithm is not satisfactory for a cluster with a slow interconnection network.

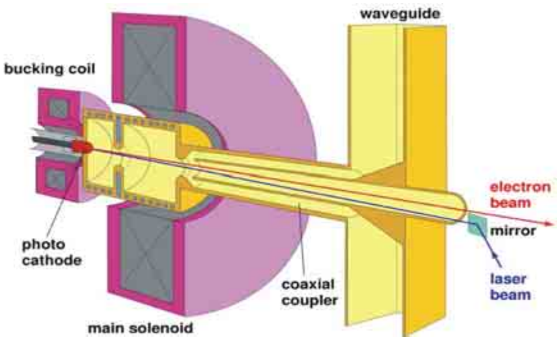
Benchmark Results

- Benchmark: electron bunch traveling within an ideally conducting tube (5 million grid points).



➡ The predicted speedup is in good agreement with the results obtained on the compute cluster.

PITZ-Injector

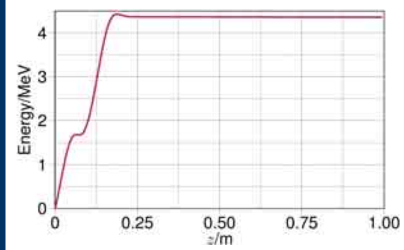


Bunch rms-radius: $\sigma_x = \sigma_y \approx 0.5 - 3.75\text{mm};\ \sigma_z \approx 2.5\text{mm}$

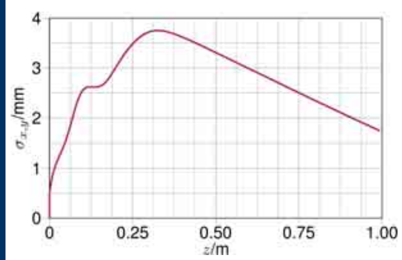
Geometry dimensions: $0.2\text{m} \times 0.2\text{m} \times 1.0\text{m}$

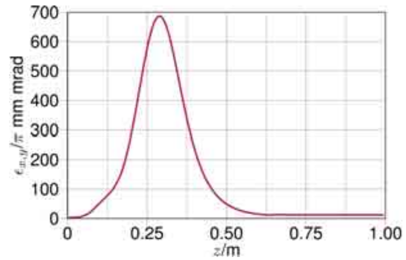
➡ **Multiscale problem: requires a high amount of computational resources.**

Results



- Fully 3D simulation using 300 million grid points and 250,000 particles.





Outline

- Motivation
- The Particle-In-Cell (PIC) Algorithm
- Parallelization and Performance Modeling for PIC
- Simulation and Performance Examples
- Conclusion

Conclusion

- Performance modeling has shown to be helpful for the construction and analysis of parallelization schemes.
- The Adaptive Bounding Box approach is well suited for the simulation of problems with very localized particle distributions.
- The Adaptive Bounding Box approach scales good on a high speed interconnection network.