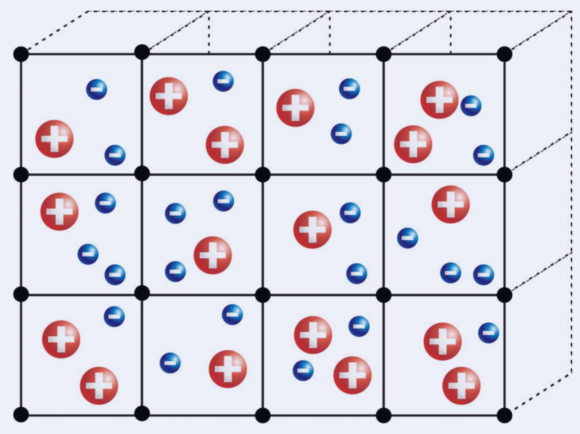
Performance and Scalability Evaluation of Particle-in-Cell Code PICADOR on CPUs and Intel Xeon Phi Coprocessors

S. Bastrakov[1], I. Meyerov[1*], I. Surmin[1], A. Bashinov[2], E. Efimenko[2],
A. Gonoskov[1,2,3], A. Korzhimanov[2], A. Larin[1], A. Muraviev[2], A. Rozanov[1]

[1] Lobachevsky State University of Nizhni Novgorod, Russia [2] Institute of Applied Physics, RAS, Russia
[3] Chalmers University of Technology, Sweden * meerov@vmk.unn.ru

Particle-in-Cell Plasma Simulation

- Two main data sets:
  - Ensemble of charged particles.
  - Grid values of electromagnetic field.
- No direct Particle-Particle interaction.
- Spatially local Particle-Grid operations.
- Important to keep efficient memory access pattern as particles move.



Field Solver

$$\nabla \times \boldsymbol{B} = \frac{4\pi}{c}\boldsymbol{j} + \frac{1}{c}\frac{\partial \boldsymbol{E}}{\partial t}$$

$$\nabla \times \boldsymbol{E} = -\frac{1}{c}\frac{\partial \boldsymbol{B}}{\partial t}$$

$\boldsymbol{E}, \boldsymbol{B}$

Field Interpolation & Field Computation

$$\boldsymbol{F}_i = q_i\left(\boldsymbol{E}(\boldsymbol{r}_i) + \frac{1}{c}\boldsymbol{v}_i \times \boldsymbol{B}(\boldsymbol{r}_i)\right)$$

Δt

Particle Push

$$\frac{d\boldsymbol{p}_i}{dt} = \boldsymbol{F}_i \qquad \frac{d\boldsymbol{r}_i}{dt} = \boldsymbol{v}_i$$

$$\boldsymbol{v}_i = \frac{\boldsymbol{p}_i}{m_i}\left(1 + \left(\frac{\boldsymbol{p}_i}{m_i c}\right)^2\right)^{-1/2}$$

$\boldsymbol{r}_i, \boldsymbol{v}_i$

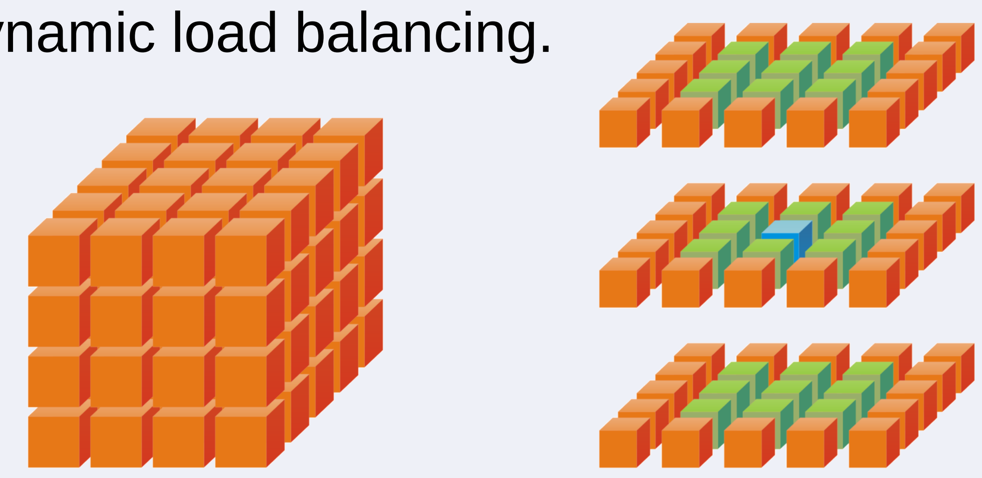
Current Deposition

$$\boldsymbol{j}(\boldsymbol{r}) = \sum_i q_i \boldsymbol{v}_i \delta(\boldsymbol{r} - \boldsymbol{r}_i)$$

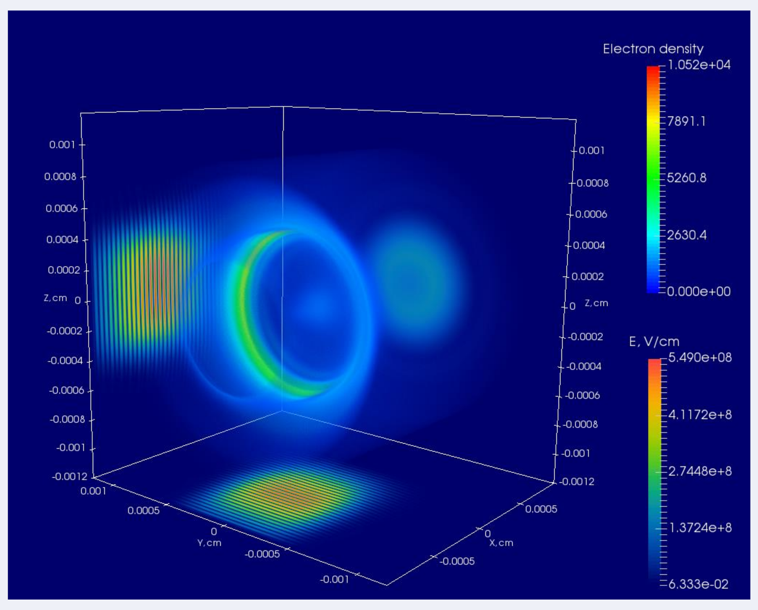
$\boldsymbol{j}$

PICADOR Code Overview

- Tool for 3D Particle-in-Cell plasma simulation.
- Heterogeneous CPU + Xeon Phi code.
- MPI + OpenMP parallelism.
- Optimized computational core, support for extensions.
- State-of-the-art numerical schemes.
- Rectilinear domain decomposition.
- MPI exchanges only between neighbors.
- Dynamic load balancing.



Application: plasma wakefield self-compression of laser pulses



Application: target normal sheath acceleration

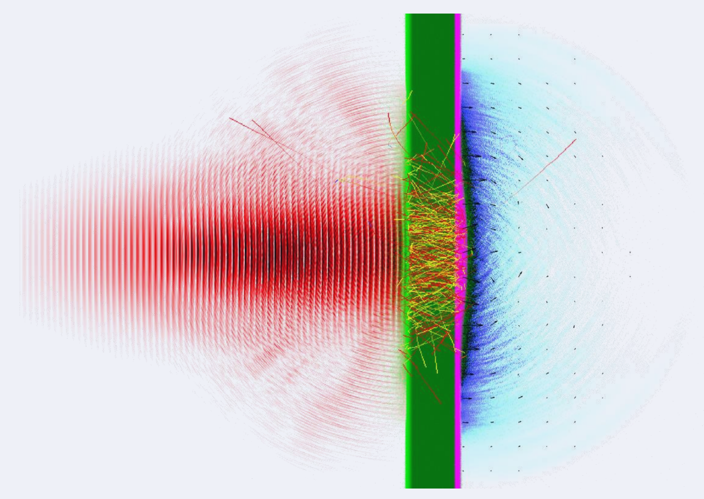
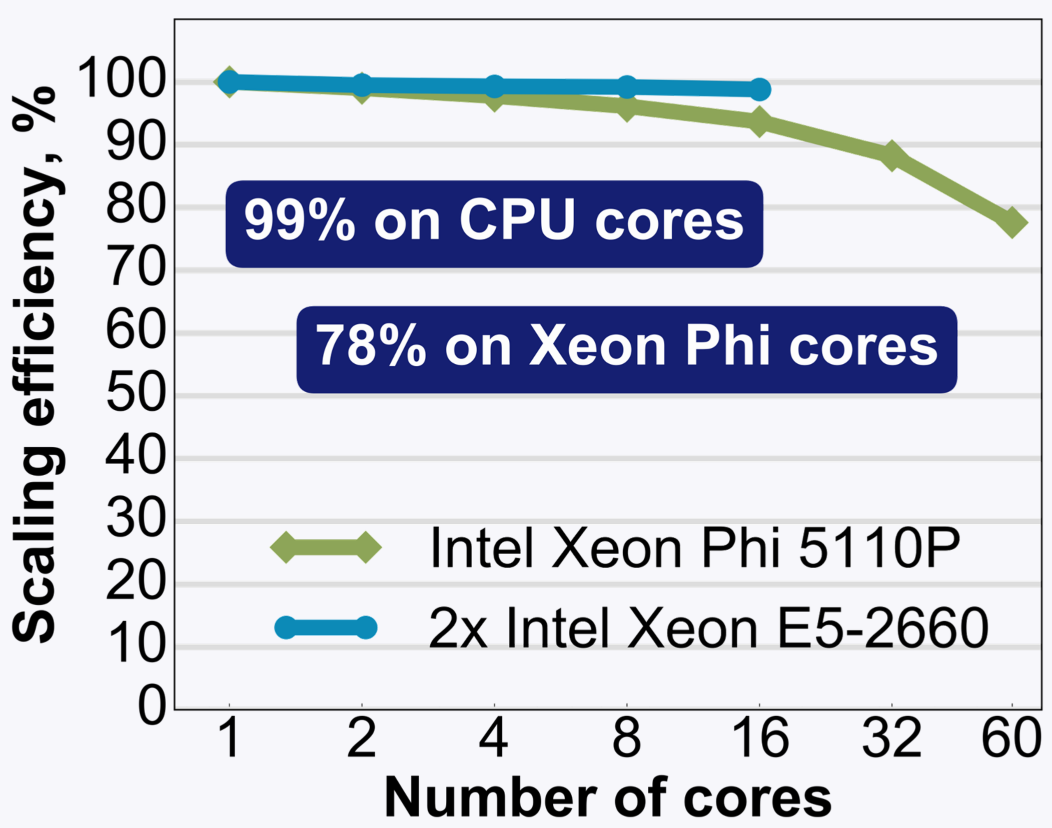


Image courtesy of Joel Magnusson, Chalmers University of Technology.

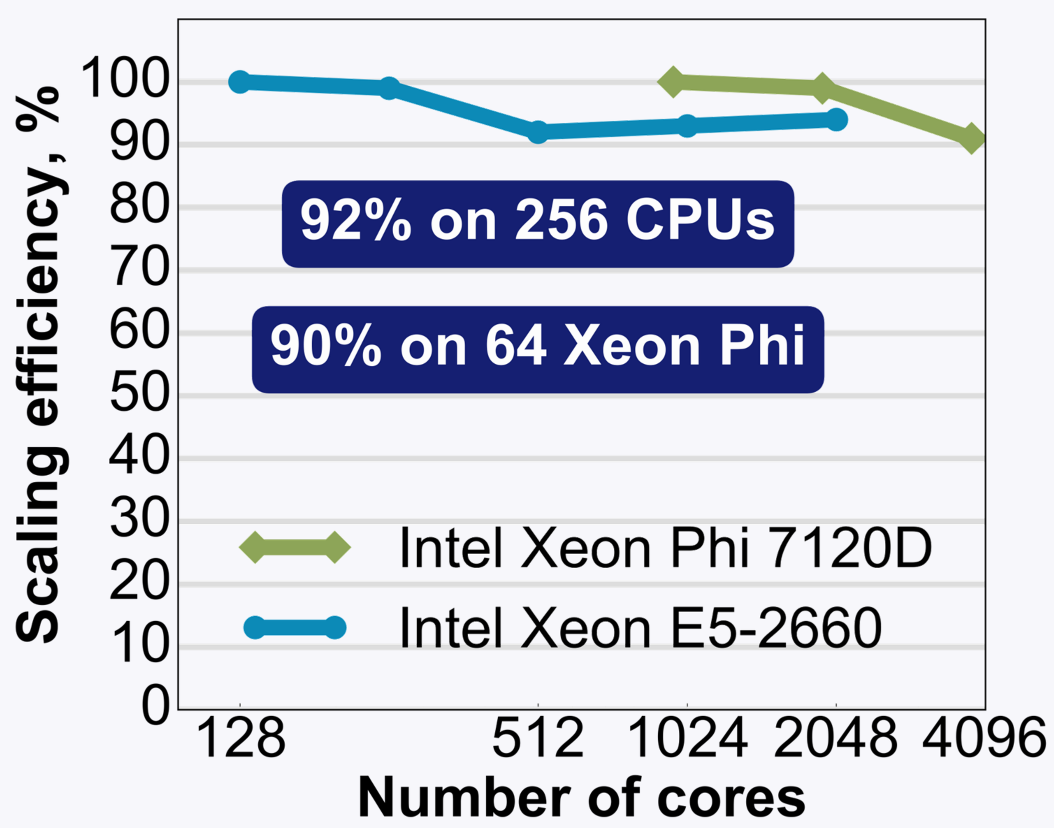
Scaling Efficiency and Performance on CPUs and Xeon Phi Coprocessors

Strong Scaling on Shared Memory*



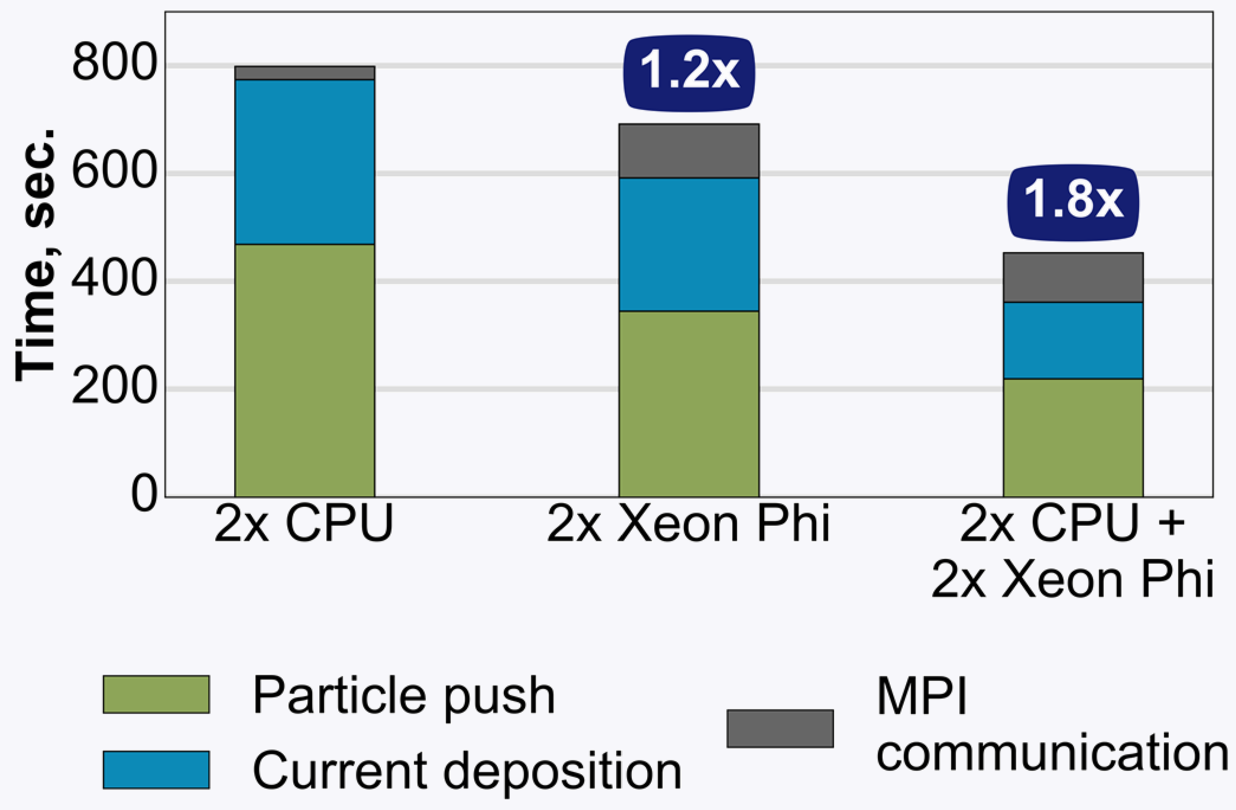
Simulation: frozen plasma benchmark.
Parameters: 40×40×40 grid, 3.2 mln. particles, first-order particle form factor.
System: *Lobachevsky Supercomputer* (UNN), 2x Intel Xeon E5-2660 + Intel Xeon Phi 5110P per node, Intel Compiler 15.0.3, Intel MPI 4.1.

Strong Scaling on Distributed Memory



Simulation: laser wakefield acceleration.
Parameters: 512×512×512 grid, 1015 mln. particles, second-order particle form factor.
System: 1) *Triolith* (NSC, Sweden), 2x Intel Xeon E5-2660 per node, Infiniband FDR, Intel Compiler 15.0.1, Intel MPI 5.0.2; 2) *RSC PetaStream*, Infiniband FDR, Intel Compiler 15.0.0, Intel MPI 5.0.3.

CPU + Xeon Phi Performance



Simulation: interaction of relativistically strong 30 fs Ti:Sa laser pulse with ionized gas jet, resulting in plasma wakefield self-compression of laser pulses.
Parameters: 256×128×128 grid, 78.5 mln. particles, first-order particle form factor, charge conserving current deposition.
System: *MVS-10P (JSC RAS)*, 2x Intel Xeon E5-2690 + 2x Intel Xeon Phi 7110X per node, Infiniband FDR, Intel C++ Compiler 14.0.1, Intel MPI 4.1.

* *I.A. Surmin, et. al. Computer Physics Communications, 2016, Vol. 202, P. 204–210.*

Roofline Performance Model on Xeon Phi. Road to KNL

Configuration for building roofline model: double precision, first-order particle form factor, 40×40×40 grid, 50 particles per cell, particle data = 64 Bytes.

Arithmetic intensity (AI) of the Particle-in-Cell core:

- AI(field interpolation + particle push) = 239 Flop / particle = 3.73 Flop / Byte.
- AI(current deposition) = 114 Flop / particle + 192 Flop / cell = 1.24 Flop / Byte.
- AI(overall) = 1.44 Flop / Byte.

Prospects on KNL:

- Expect a 3x increase in a single-core performance with the same SIMD width to translate into a similar increase of code performance without a lot of effort.
- Efficient vectorization of the current deposition stage will be still problematic.
- A growing performance gap between CPU and KNL will require a special load balancing scheme for heterogeneous CPU + KNL runs.

Roofline Performance Model on Xeon Phi 7110X