Particle-in-Cell Plasma Simulation on CPUs, GPUs and Xeon Phi Coprocessors

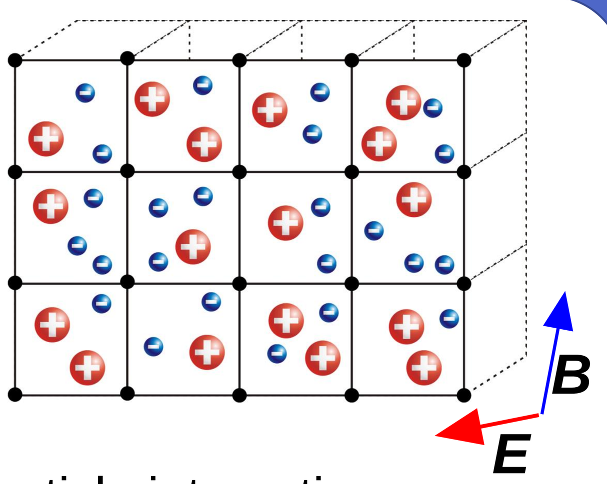
Sergey Bastrakov[1], Iosif Meyerov[1*], Igor Surmin[1], Evgeny Efimenko[2], Arkady Gonoskov[2], Alexander Malyshev[1], Mikhail Shiryaev[1]**

[1] N.I. Lobachevsky University of Nizhni Novgorod, Russia, [2] Institute of Applied Physics, RAS, Nizhni Novgorod, Russia

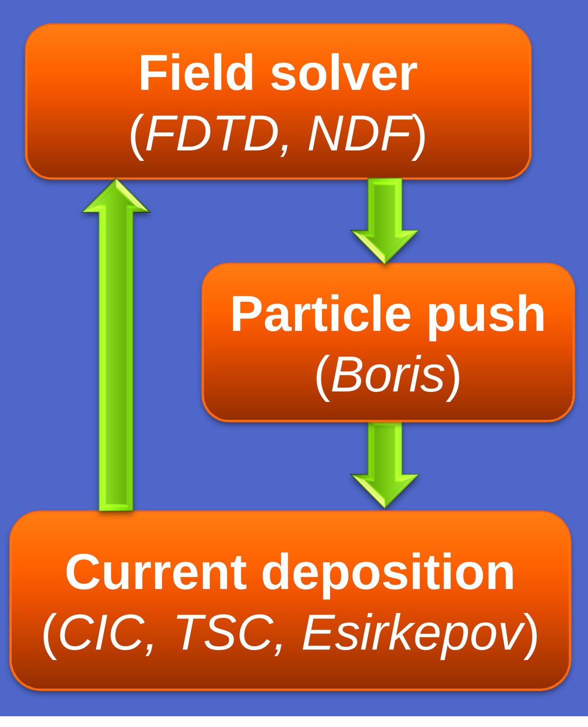
* meerov@vmk.unn.ru ** arkady.gonoskov@gmail.com

Particle-in-Cell Plasma Simulation

- Two sets of data:
  - **Ensemble** of charged particles.
  - **Grid** values of electromagnetic field and current density.
- No direct Particle-Particle interaction.
- Particle-Grid operations are spatially local.
- Memory access pattern changes as particles move. Need reordering of particles to keep efficient memory access pattern.



Computational scheme



PICADOR Overview

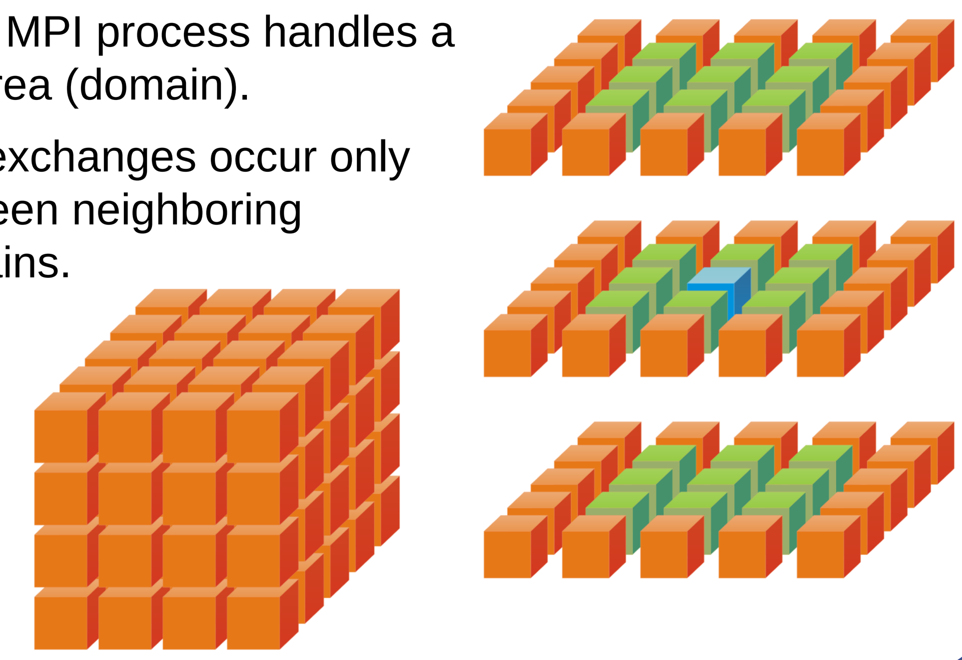
PICADOR* is a tool for three-dimensional plasma simulation using the Particle-in-Cell (PIC) method. It is capable of running on CPUs, GPUs and Intel Xeon Phi coprocessors, and supports dynamic load balancing.

Other features of PICADOR include FDTD and NDF field solvers, Boris particle pusher, CIC and TSC particle form factors, Esirkepov current deposition, ionization, moving frame.

* S. Bastrakov, R. Donchenko, A. Gonoskov, E. Efimenko, A. Malyshev, I. Meyerov, I. Surmin, Particle-in-cell plasma simulation on heterogeneous cluster systems, Journal of Computational Science, 3 (2012), 474-479.

Domain decomposition

- Each MPI process handles a subarea (domain).
- MPI exchanges occur only between neighboring domains.

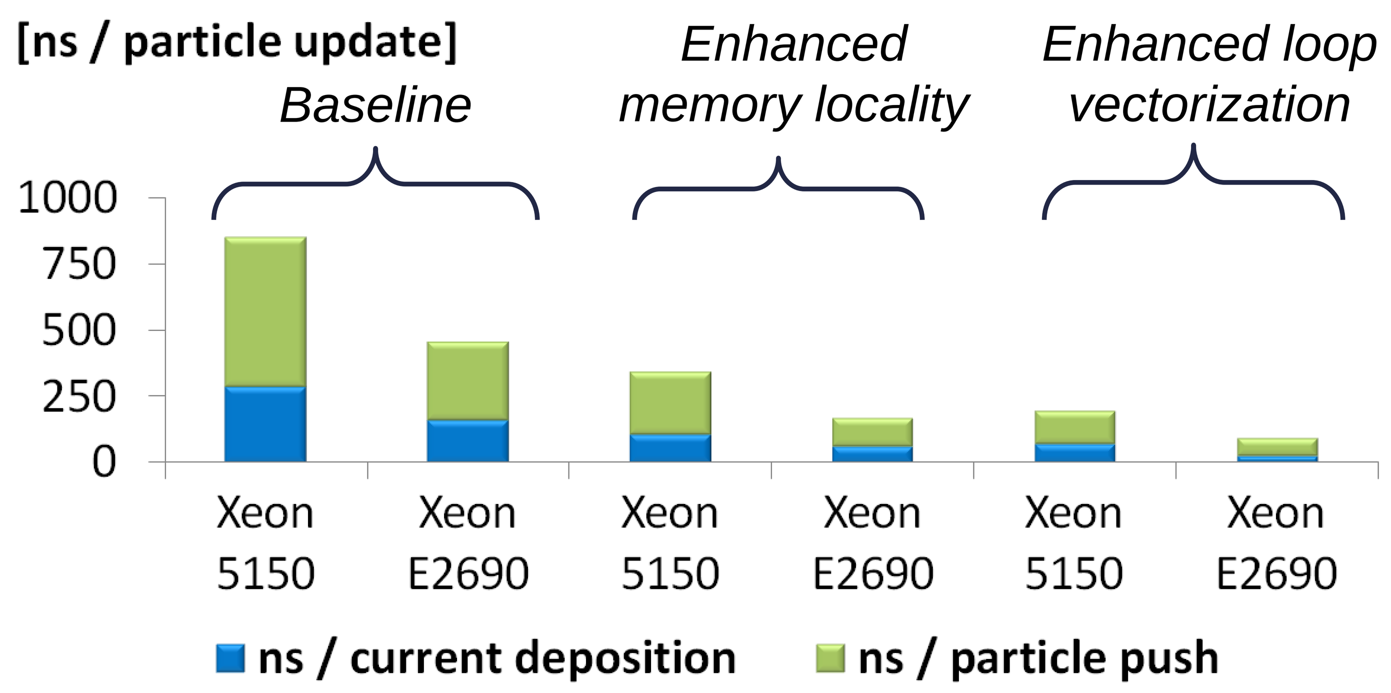


Performance and Scaling on CPUs

Implementation overview

- The main performance concern is **memory access pattern** during field interpolation and current deposition.
- A separate array of particles per each cell.
- Particles are processed cell by cell.
- *Enhanced memory locality*: while processing particles in a cell, preload field values / accumulate currents locally.
- We assist compiler to *vectorize particles loops* via inlining, compiler directives, loop subdivision and tiling.
- Still, field component scatter in Yee grid hinders efficient memory access in vectorized field interpolation.
- OpenMP threads handle particles in different cells in parallel.

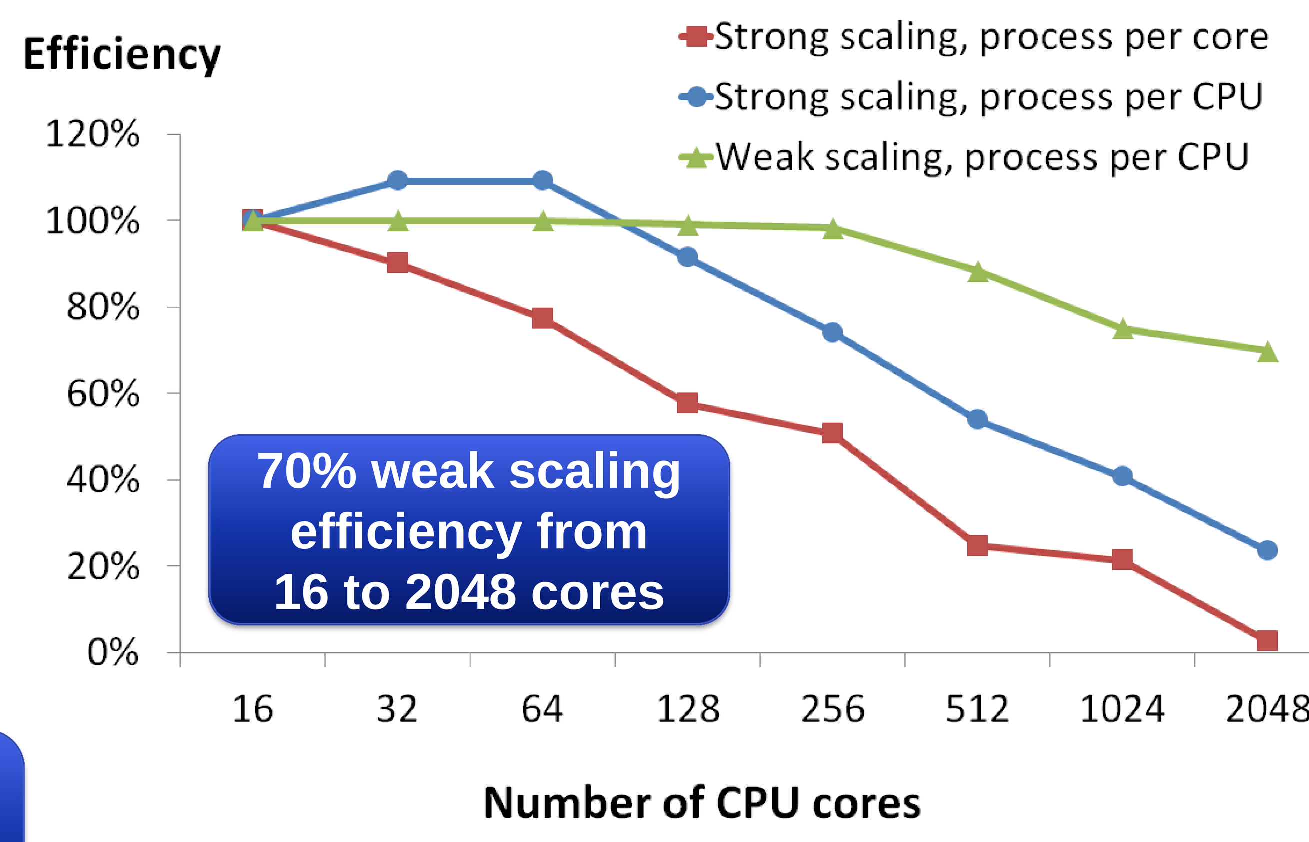
Single-core CPU performance



Speedup 4x to 7x over baseline.
The optimized single-core performance on Xeon E2690: 95 ns / particle update, 26% of peak performance

Benchmark: 10^8 particles, 2.1×10^6 cells, CIC particle form factor, double precision.
Hardware: Intel Xeon 5150 at UNN, Intel Xeon E2690 at MVS-10P (JSC RAS).

Scaling efficiency



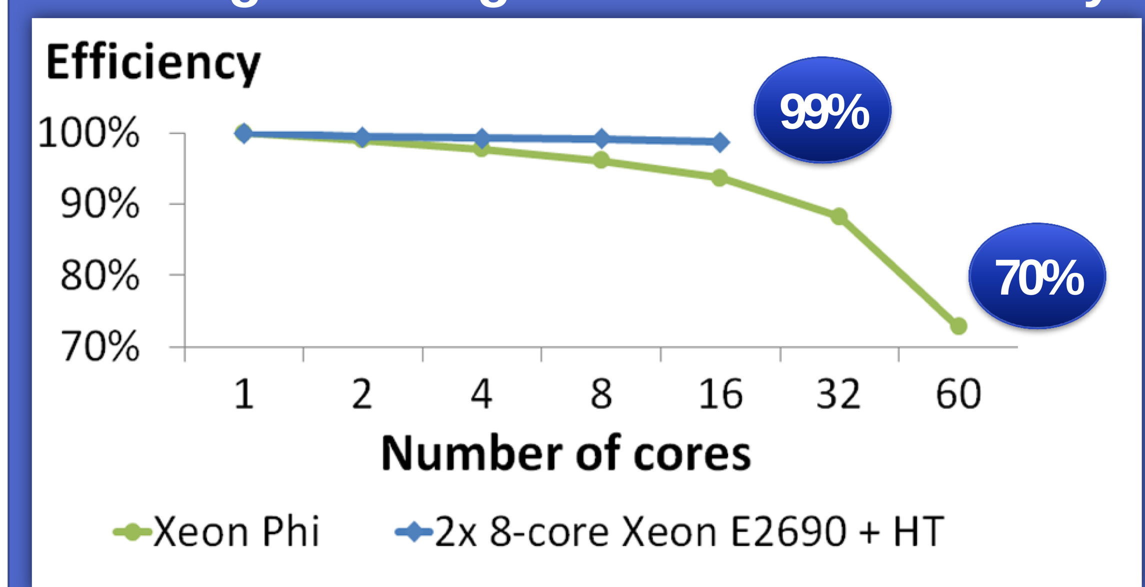
Benchmark: CIC particle form factor, double precision. Strong scaling: 10^8 particles, 2.1×10^6 cells. Weak scaling: 6.5×10^6 particles, 1.3×10^5 cells per core.
Hardware: Intel Xeon E2690, Infiniband at MVS-10P (JSC RAS).

Performance on Xeon Phi

Implementation overview

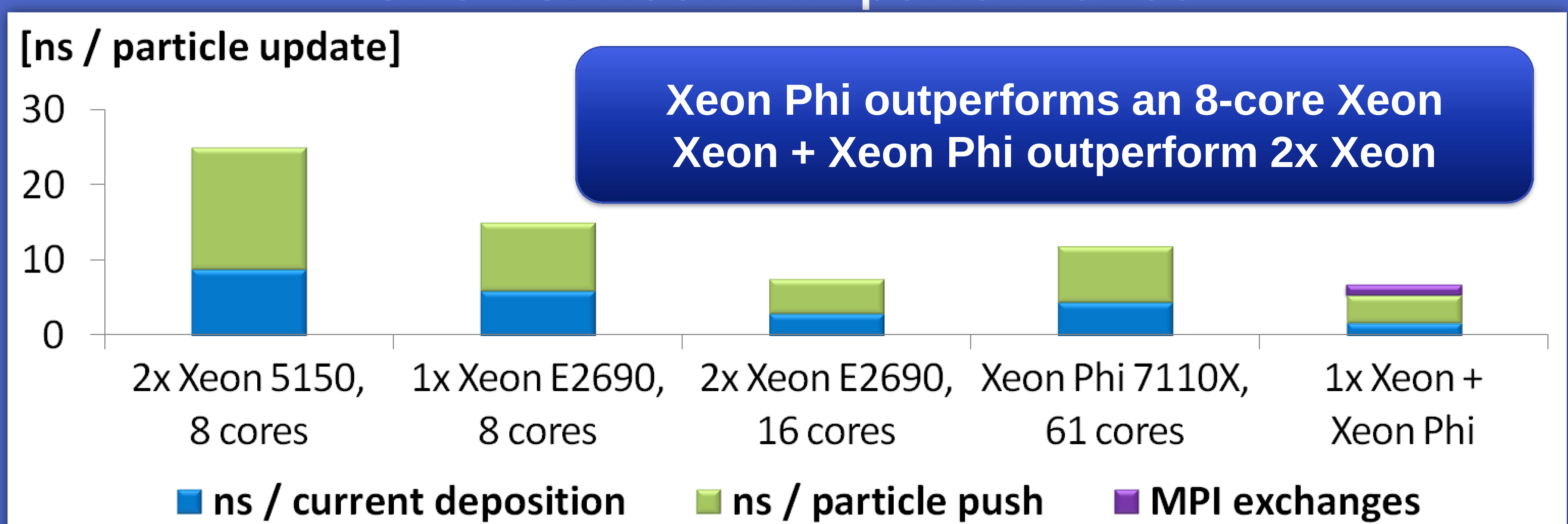
- We use Xeon Phi in native and symmetric modes.
- The same C++ code for CPUs and Xeon Phi.
- General performance considerations are similar to CPUs with a stronger emphasis on vectorization and scaling efficiency.
- Use 4 threads per core.

Strong scaling on shared memory



Benchmark: Langmuir plasma oscillations, 10^8 particles, 2.1×10^6 cells, CIC particle form factor, double precision.
Hardware: Intel Xeon E2690, Intel Xeon Phi 7110X, at MVS-10P (JSC RAS).

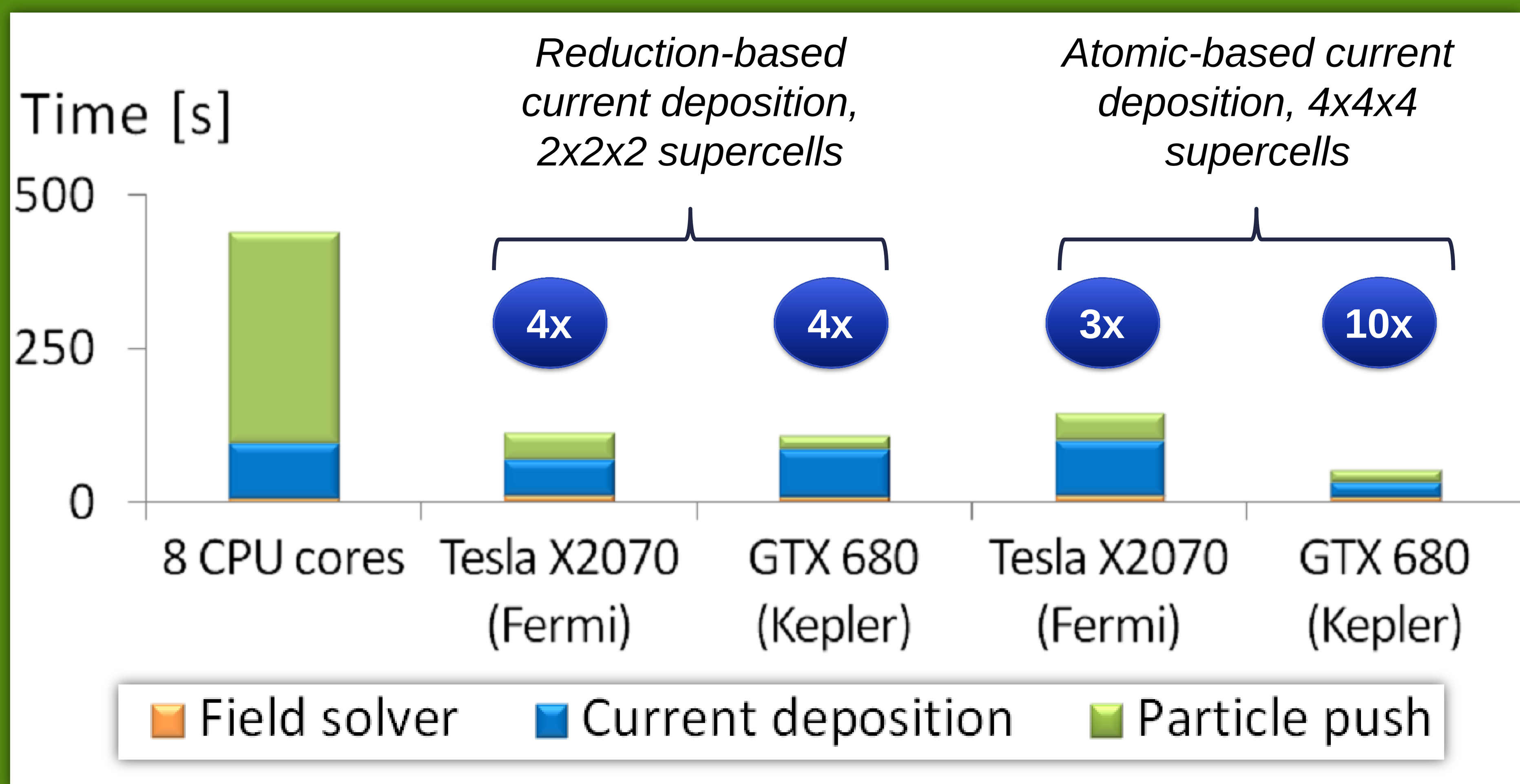
CPU vs. Xeon Phi performance



Xeon E2690: 15 nanoseconds per particle update
Xeon Phi: 12 nanoseconds per particle update
Xeon + Xeon Phi: 6.7 nanoseconds per particle update

Benchmark: Langmuir plasma oscillations, 10^8 particles, 2.1×10^6 cells, CIC particle form factor, double precision.
Hardware: Intel Xeon 5150 at UNN; Intel Xeon E2690, Intel Xeon Phi 7110X, Infiniband at MVS-10P (JSC RAS).

Performance on GPUs



Benchmark: 40×40×40 grid, 50 particles per cell, 100 time steps, **single precision** (SP).
Hardware: Intel Xeon L5630; NVIDIA Tesla X2070 (Fermi, 448 cores), 1030 GFLOPS in SP; GTX 680 (Kepler GK104, 1536 cores), 3090 GFLOPS in SP.

Summary and Future Work

PICADOR is developed and used by the HPC Center of University of Nizhni Novgorod and the Institute of Applied Physics of Russian Academy of Sciences for simulation of laser-matter interaction. The code architecture is extendable in terms of additional stages and devices and is capable of using contemporary heterogeneous cluster systems with CPUs, GPUs and Intel Xeon Phi coprocessors. The performance and scaling efficiency are competitive with other implementations.

The future prospects include better load balancing between CPUs, GPUs and Xeon Phi, further optimization of GPU and Xeon Phi implementations, development and optimization of additional modules to allow solving a larger set of problems.