WarpX: Toward Exascale Modeling of Plasma Particle Accelerators on CPU and GPU

Maxence Thévenet
LBNL
Berkeley, California
mthevenet@lbl.gov

Jean-Luc Vay
LBNL
Berkeley, California
jlvay@lbl.gov

Ann Almgren
LBNL
Berkeley, California
ASAlmgren@lbl.gov

Diana Amorim
LBNL
Berkeley, California
ldianaamorim@lbl.gov

John Bell
LBNL
Berkeley, California
jbbell@lbl.gov

Axel Huebl
LBNL
Berkeley, California
axelhuebl@lbl.gov

Revathi Jambunathan
LBNL
Berkeley, California
rjambunathan@lbl.gov

Rémi Lehe
LBNL
Berkeley, California
rlehe@lbl.gov

Andrew Myers
LBNL
Berkeley, California
atmyers@lbl.gov

Jaehong Park
LBNL
Berkeley, California
jaehongpark@lbl.gov

Olga Shapoval
LBNL
Berkeley, California
oshapoval@lbl.gov

Weiqun Zhang
LBNL
Berkeley, California
WeiqunZhang@lbl.gov

Lixin Ge
SLAC
Menlo Park, California
lge@slac.stanford.edu

Mark Hogan
SLAC
Menlo Park, California
hogan@slac.stanford.edu

Cho Ng
SLAC
Menlo Park, California
cho@slac.stanford.edu

David Grote
LLNL
Livermore, California
grote1@llnl.gov

ACM Reference Format:
Maxence Thévenet, Jean-Luc Vay, Ann Almgren, Diana Amorim, John Bell, Axel Huebl, Revathi Jambunathan, Rémi Lehe, Andrew Myers, Jaehong Park, Olga Shapoval, Weiqun Zhang, Lixin Ge, Mark Hogan, Cho Ng, and David Grote. 2019. WarpX: Toward Exascale Modeling of Plasma Particle Accelerators on CPU and GPU. In *Proceedings of Supercomputing 2019 (SC19)*. ACM, New York, NY, USA, 3 pages.

Particle accelerators are a vital part of the DOE-supported infrastructure of discovery science and university and private-sector applications, and have a broad range of benefits to industry, security, energy, the environment and medicine. To take full advantage of their societal benefits, however, we need game-changing improvements in the size and cost of accelerators. Plasma-based particle accelerators stand apart in their potential for these improvements. Turning this from a promising technology into mainstream scientific tools depends critically on high-performance, high-fidelity modeling of complex processes that develop over a wide range of space and time scales.

As part of DOE's Exascale Computing Project [3], a team from Lawrence Berkeley National Laboratory, in collaboration with teams from SLAC National Accelerator Laboratory and Lawrence Livermore National Laboratory, is developing a new powerful plasma accelerator simulation tool. The new software will harness the power of future exascale supercomputers for the exploration of outstanding questions in the physics of acceleration and transport of particle beams in chains of plasma channels. This will benefit the ultimate goal of compact and affordable high-energy physics colliders, and many spinoff applications of plasma accelerators along the way.

We are combining three major software components (Warp, AMReX and PICSAR) into the new software tool WarpX[1][2] that will be tuned for running efficiently at scale on exascale supercomputers.

SC19, November 2019, Denver, Colorado USA
2019.

[1]This work was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of two U.S. Department of Energy organizations (Office of Science and the National Nuclear Security Administration) responsible for the planning and preparation of a capable exascale ecosystem, including software, applications, hardware, advanced system engineering, and early testbed platforms, in support of the nation's exascale computing imperative. Used resources of the National Energy Research Scientific Computing Center.
[2]PI of the WarpX project: Jean-Luc Vay

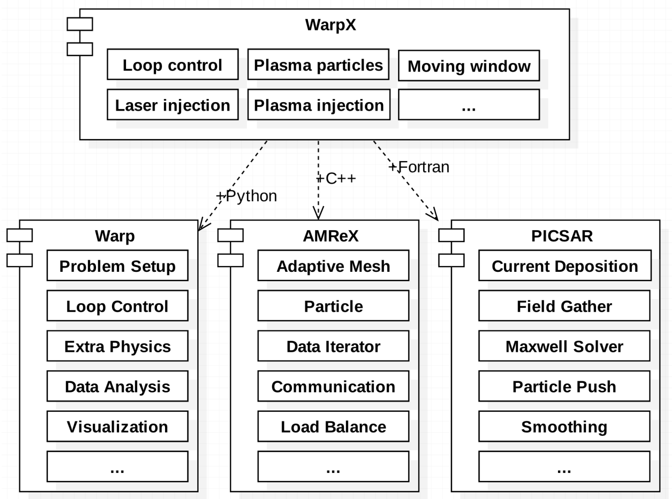

Figure 1: UML diagram of WarpX. In addition to WarpX's own source code ('WarpX-Source'), functionalities are provided by three packages: AMReX for the handling of AMR, communication and load balancing, PICSAR for the low level individual PIC functionalities, and Warp (optional) for extra physics packages, alternate user interface and control.

A UML diagram of the WarpX application is given in Fig. 1. Warp [4] is a framework for the modeling of plasma, beam and particle accelerators using state-of-the-art parallel Particle-In-Cell (PIC) methods. The Adaptive Mesh Refinement (AMR) library AMReX [1] is a robust implementation of the AMR methodology for various applications. The novel library PICSAR [2], developed in collaboration with researchers from CEA Saclay in France, implements the elementary PIC operations with extensive optimizations for many-core architectures. These three packages (Warp [4], AMReX [1] and PICSAR [2]) are orchestrated together with WarpX's source code ('WarpX-Source') as follows:

- Warp [4] is a pre-existing PIC code with a Python interface and FORTRAN subroutines for fast number crunching. Warp's extensive collection of functionalities can be leveraged by WarpX, when running with Python as the front-end (optional): the Warp framework provides a Python interface with modules for code control, user steering and additional physics (not included in WarpX-Source, AMReX or PICSAR) and diagnostic packages.
- The PICSAR library [2] contains FORTRAN subroutines (originally from Warp) for elemental Particle-In-Cell operations at the innermost loop on grid and particle data, i.e. charge and current deposition from the particles to the grid, Maxwell solver, field gather from the grid to the particles and particle pusher. These subroutines have been (and continue to be) highly optimized for new multicore architectures such as Intel KNL.
- The PICSAR subroutines are called within intermediate loops for each set of grids and particles of the block-structured AMR hierarchy by the AMReX library (C++) [1], which also handles intra-node (OpenMP) and inter-node (MPI) parallelism, load balancing and parallel I/O on those data.
- WarpX-Source is a set of C++, FORTRAN and Python subroutines for interfacing Warp, AMReX and PICSAR. WarpX-Source does not just stitch together the three other components, but orchestrates the workflow at the core of the main loop level. WarpX-Source contains an alternate C++ driver for convenience in development, testing, profiling, and running on systems where Python support is problematic. It also contains some recoded pre-existing Warp functionalities, as needed for compatibility with AMReX.

The main algorithm that is used in WarpX is the electromagnetic Particle-In-Cell method, where charged macroparticles are pushed using the Newton-Lorentz equations (with a leapfrog pusher), interacting via self-electromagnetic fields that are solved on a hierarchy of AMR block-structured grids. Interpolations using linear, quadratic or cubic splines are used to deposit the macroparticles' currents on the grids and to gather the electromagnetic fields from the grids onto the macroparticles. The Maxwell's equations are solved using either a finite-difference time-domain (FDTD) solver or an FFT-based Pseudo-Spectral Analytical Time-Domain (PSADT) solver.

The mesh refinement method that was implemented in WarpX follows the method that had been implemented and validated in Warp [8], based on the following principles: i) avoidance of spurious effects from mesh refinement, or minimization of such effects; ii) user controllability of the spurious effects' relative magnitude; iii) simplicity of implementation. The two main generic issues that were identified in earlier work are: a) spurious self-force on macroparticles close to the mesh refinement interface [5, 7]; b) reflection (and possible amplification) of short wavelength electromagnetic waves at the mesh refinement interface [6]. The two effects are due to the loss of translation invariance introduced by the asymmetry of the grid on each side of the mesh refinement interface. The implementation that was adopted in WarpX mitigates both spurious effects.

Numerical dispersion is always present in finite-difference algorithms. With typical second order accurate solvers (the most common being the so-called *Yee* solver[9]), keeping the numerical dispersion at an acceptable level requires strongly over-resolving the wavelength of the incoming laser and scattered radiation, which leads to a very large number of grid cells. By contrast, spectral algorithms enable solvers with very high spatial order. Combining spectral algorithms with analytical integration of the equations over one time-step (assuming that the source term is constant over the time step, a core Particle-In-Cell assumption), they lead to the Pseudo-Spectral Analytical Time-Domain (PSATD) algorithm that exhibits no numerical dispersion at any wavelength or angle, and has no Courant. PICSAR offers an optimized PSATD solver that is used in WarpX. Due to the finite speed of light, this implementation relies on local FFTs and proves to be scalable on large supercomputers like Mira at ALCF.

WarpX has been used for production simulations on CPUs, and has been ported to GPUs, showing good scalings for uniform plasma simulations on half of the Summit supercomputer at Oak Ridge Leadership Computing Facility (OLCF). The whole PIC loop runs on the GPU, to avoid time-consuming data transfer between CPU and GPU as much as possible. WarpX strategy follows the AMReX strategy, consisting in using extensively the CUDA Managed memory, provided an abstraction level with `amrex::ParallelFor`, similar to `Kokkos` and `RAJA` for instance, and using the C++ `Thrust` library.

Finally, the WarpX Exascale project combines state-of-the-art libraries for particle-in-cell algorithm (PICSAR) and adaptive mesh refinement (AMReX) along with novel algorithms to explore the physics of plasma accelerators and support experimental efforts to build a high-energy physics collider. The code shows good scalings on the NERSC supercomputer Cori KNL without mesh refinement, and teams are currently dedicated to enable mesh refinement to increase performances as well as fidelity.

REFERENCES

[1] [n. d.]. AMReX. https://ccse.lbl.gov/AMReX.
[2] [n. d.]. PICSAR. https://picsar.net.
[3] [n. d.]. U.S. DOE Exascale Computing Project. https://www.exascaleproject.org.
[4] [n. d.]. Warp. http://warp.lbl.gov.
[5] Phillip Colella and Peter C Norgaard. 2010. Controlling Self-Force Errors At Refinement Boundaries For Amr-Pic. *J. Comput. Phys.* 229, 4 (feb 2010), 947–957. https://doi.org/10.1016/J.Jcp.2009.07.004
[6] J.-L. Vay. 2001. An Extended Fdtd Scheme For The Wave Equation: Application To Multiscale Electromagnetic Simulation. *J. Comput. Phys.* 167, 1 (feb 2001), 72–98.
[7] J.-L. Vay, P Colella, P Mccorquodale, B Van Straalen, A Friedman, and D. P. Grote. 2002. Mesh Refinement For Particle-In-Cell Plasma Simulations: Applications To And Benefits For Heavy Ion Fusion. *Laser And Particle Beams* 20, 4 (dec 2002), 569–575. https://doi.org/10.1017/S0263034602204139
[8] J.-L. Vay, D P Grote, R H Cohen, and A Friedman. 2012. Novel methods in the particle-in-cell accelerator code-framework warp. *Computational Science and Discovery* 5, 1 (2012), 014019 (20 pp.).
[9] Ks Yee. 1966. Numerical Solution Of Initial Boundary Value Problems Involving Maxwells Equations In Isotropic Media. *Ieee Transactions On Antennas And Propagation* Ap14, 3 (1966), 302–307.