Physics of Plasmas

AIP Publishing

RESEARCH ARTICLE | FEBRUARY 09 2021

Modeling of a chain of three plasma accelerator stages with the WarpX electromagnetic PIC code on GPUs

Special Collection: Building the Bridge to Exascale Computing: Applications and Opportunities for Plasma Science

J.-L. Vay ✉ ; A. Huebl ; A. Almgren; L. D. Amorim ; J. Bell; L. Fedeli; L. Ge; K. Gott; D. P. Grote ; M. Hogan; R. Jambunathan; R. Lehe; A. Myers; C. Ng; M. Rowan; O. Shapoval; M. Thévenet; H. Vincenti; E. Yang; N. Zaïm; W. Zhang; Y. Zhao ; E. Zoni

Phys. Plasmas 28, 023105 (2021)
https://doi.org/10.1063/5.0028512
CHORUS

Modeling of a chain of three plasma accelerator stages with the WarpX electromagnetic PIC code on GPUs

Cite as: Phys. Plasmas **28**, 023105 (2021); doi: 10.1063/5.0028512
Submitted: 4 September 2020 · Accepted: 11 January 2021 ·
Published Online: 9 February 2021

J.-L. Vay,[1,a)] A. Huebl,[1] A. Almgren,[1] L. D. Amorim,[1] J. Bell,[1] L. Fedeli,[2] L. Ge,[3] K. Gott,[1] D. P. Grote,[4] M. Hogan,[3] R. Jambunathan,[1] R. Lehe,[1] A. Myers,[1] C. Ng,[3] M. Rowan,[1] O. Shapoval,[1] M. Thévenet,[5] H. Vincenti,[2] E. Yang,[1] N. Zaïm,[2] W. Zhang,[1] Y. Zhao,[1] and E. Zoni[1]

AFFILIATIONS

[1]Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA
[2]LIDYL, CEA-Université Paris-Saclay, CEA Saclay, 91 191 Gif-sur-Yvette, France
[3]SLAC National Accelerator Laboratory, Menlo Park, California 94025, USA
[4]Lawrence Livermore National Laboratory, Livermore, California 94550, USA
[5]Deutsches Elektronen Synchrotron (DESY), Hamburg, Hamburg 22607, Germany

Note: This paper is part of the Special Collection: Building the Bridge to Exascale Computing: Applications and Opportunities for Plasma Science.

[a)]**Author to whom correspondence should be addressed:** jlvay@lbl.gov

ABSTRACT

The fully electromagnetic particle-in-cell code WarpX is being developed by a team of the U.S. DOE Exascale Computing Project (with additional non-U.S. collaborators on part of the code) to enable the modeling of chains of tens to hundreds of plasma accelerator stages on exascale supercomputers, for future collider designs. The code is combining the latest algorithmic advances (e.g., Lorentz boosted frame and pseudo-spectral Maxwell solvers) with mesh refinement and runs on the latest computer processing unit and graphical processing unit (GPU) architectures. In this paper, we summarize the strategy that was adopted to port WarpX to GPUs, report on the weak parallel scaling of the pseudo-spectral electromagnetic solver, and then present solutions for decreasing the time spent in data exchanges from guard regions between subdomains. In Sec. IV, we demonstrate the simulations of a chain of three consecutive multi-GeV laser-driven plasma accelerator stages.

I. INTRODUCTION

Of the possible applications of plasma-based particle accelerators, the development of a plasma-based collider for high-energy physics studies is the most challenging.[1] Fast and accurate simulation tools are required to study the physics toward configurations that enable the production and acceleration of very small beams with low energy spread and emittance preservation over long distances, as required for a collider. The fully electromagnetic particle-in-cell (PIC) code WarpX[2] is being developed by a team of the U.S. DOE Exascale Computing Project[3] (with additional non-U.S. collaborators on part of the code) to enable the modeling of chains of tens to hundreds of plasma accelerator stages on exascale supercomputers, for future collider designs. The code is combining the latest algorithmic advances (e.g., Lorentz boosted frame[4] and pseudo-spectral Maxwell solvers[5]) with mesh refinement and runs on the latest computer processing unit (CPU) and graphical processing unit (GPU) architectures. The first version of the code, together with its first application to the modeling of plasma acceleration with mesh refinement, was reported in a previous publication.[6] A follow-up publication[7] reported on the implementation and effect of subcycling on simulations with mesh refinement, together with a discussion on load balancing.

In this paper, we summarize the strategy that was adopted to port WarpX to GPUs. In Sec. III, we report on the weak parallel scaling of the pseudo-spectral electromagnetic solver and then present solutions for decreasing the time spent in data exchanges from guard regions between subdomains. In Sec. IV, we present simulations of three consecutive multi-GeV laser-driven plasma accelerator stages on GPUs that confirm the accuracy of the new version of the code, using either the finite-difference or the pseudo-spectral electromagnetic solver.

II. GPU PORTABILITY STRATEGY AND IMPLEMENTATION

WarpX is now a modern C++14 application. It follows a single-source programing approach,[8] meaning that individual algorithms are implemented once and compiled for a targeted hardware through syntax and library abstractions. These abstractions are provided through the AMReX library[9] in the form of generic interfaces to parallel-for and reduction primitives, which, in turn, spawn parallel compute kernels.

Concrete numerical algorithms, such as the particle push, are defined and implemented as a callable object, that is, passed to a parallel primitive. A parallel primitive evaluates the callable object over the index-space that spans the data, i.e., particle and cell indices. WarpX often implements its numerical algorithms in-place via unnamed functions, so-called lambdas. Lambda-based algorithm design is compact and often used by contemporary C++ portability frameworks.[10–12] All computer operations in WarpX are implemented in the aforementioned manner, which enables complete GPU acceleration and avoids host-device data transfers for the critical path of the application.[13]

Unnoticed by the WarpX developer, AMReX's low-level, on-node primitives are implemented with accelerated programing models, most prominently OpenMP 3.1+ for CPU and CUDA 9.0+ for Nvidia GPU support. Consequently, ongoing activities that will enable Intel GPU support (through SYCL/DPC++[12]) and AMD GPU support (through HIP[14]) are causing little to no change to numerical algorithms implemented in WarpX.

Besides providing a performance-portability layer, AMReX also unifies data structures (layout) and containers (data management) for particle and mesh structures. Data containers provide the basis for iterating and updating field and particle values on CPUs and GPUs. While computing on a multi-core CPU, work sharing between individual threads is achieved by tiling the domain in these containers further, which can lead to cache re-use because elements in a tile are grouped spatially close together. An individual thread operates on a tile of data, which aids potential compiler-guided vectorization and avoids cache conflicts between threads.

In contrast, the GPU strategy currently assigns individual cells and particles to GPU threads without yet exploiting explicit cache blocking algorithms such as supercell structures.[15] For example, current deposition is performed via hardware-accelerated atomic additions into global device memory. Even without cache blocking particle structures, data locality can be increased. In particular, repeated sorting of particles into spatially close buckets over time was explored and has shown to enable a simulation speedup of up to nearly eightfold over unsorted particles.[16]

Besides the aforementioned on-node composition of particle and cell data elements, data containers also serve the purpose to domain-decompose a large simulation volume across nodes of a computing cluster. As in on-node tiling, overlapping halo (or guard) regions are managed with container functions that can exchange and populate values from neighboring sub-domains. Abstracting such functionality via AMReX containers is convenient, since complex functionality such as load balancing, changing execution strategies with number of domain-decomposed boxes per device, and organizing data between multiple mesh-refinement levels can be optimized by domain experts who work on abstract modules. For example, one can optimize cost functions and space filling curves for load balancing, numerical details of an algorithm, parallel execution strategy and tiling, and among other aspects independently and in different locations of the software stack.

III. WEAK PARALLEL SCALING OF THE PSEUDO-SPECTRAL ELECTROMAGNETIC SOLVER

In this section, we explore the weak parallel scaling of the pseudo-spectral analytical time-domain (PSATD) solver on the Oak Ridge Leadership Facility (OLCF) computer cluster Summit[17] using one fast Fourier transform (FFT) per GPU. The computer hardware on each Summit node are two IBM POWER9 CPUs and six NVIDIA V100 (16 GB) GPUs, connected with NVLink. We then discuss the results and possible paths for improvements.

The standard particle-in-cell (PIC) method[18] uses the finite-difference-time-domain (FDTD or "Yee" scheme)[19] to solve Maxwell's equations. It is based on second-order, centered, finite-differences that are staggered in space and time ("leapfrog" arrangement) and is, thus, time-reversible and energy-conserving, as well as local and easily parallelizable with excellent scaling to many nodes. Its low order in space and time finite-differencing leads, however, to significant numerical dispersion, which can be especially detrimental to the accuracy of the simulations of lasers and relativistic charged particle beams.

Variations of FDTD have been introduced,[20–24] based on the usage of more general nonstandard finite difference (NFD) stencils.[25,26] The NFD stencils that are used add extra points in the direction of the finite-differencing, in the plane, that is, perpendicular to it, or both. The main effect of these solvers is to either enable larger time steps at which the numerical dispersion vanishes along the axis of propagation of the laser or charged particle beam, or to reduce the effects of numerical dispersion by tailoring the velocity of the shortest wavelengths to be superluminal rather than subluminal (with the physical speed of light in vacuum as the reference). The order of the differencing is kept to two, and there is still some numerical dispersion effects that can affect the accuracy of simulations for waves propagating at an angle from the grid axes.

Reducing numerical dispersion and its effects requires the use of higher-order stencils, whose numerical cost increases with the order. Beyond a given order, it becomes more efficient to use fast Fourier transforms (FFTs) rather than direct finite-differencing, as its cost is independent of the order.[27] An added benefit of using FFTs is that the linear Maxwell's equations can then be solved analytically over one time step, assuming that the source term (particles' currents) is constant (a key approximations in PIC codes).[28] The resulting algorithm—pseudo-spectral analytic time-domain (PSATD)—has no numerical dispersion at any wavelength and angle when used in its standard form with infinite order k-space spatial derivatives.

At each time step, forward FFTs of each electromagnetic field components and source terms are computed, the fields updated to the next time step using Maxwell's equations in Fourier space, and then transformed back to configuration space with backward FFTs. Since parallel FFTs are computationally costly on a large number of nodes, domain decomposition with local FFTs on subdomains is used with PSATD, exchanging guard cells data between neighboring subdomains at each time step. Indeed, it was shown in previous work[29] that domain decomposition is even possible with PSATD at the infinite order, at the expense of truncation errors[30] from the use of a finite number of guard cells. In that case, a relatively large number of guard cells are needed to minimize the magnitude of the truncation errors, reducing the efficiency of parallel scaling to many compute nodes. However, the magnitude of the errors can be dramatically reduced by using an equivalent stencil at finite—but still very high—order,[30] enabling much better parallel scaling.[5,31] Furthermore, reduced precision data types can lead to additional speedups, as explored later.

The use of domain decomposition with local FFTs for PSATD is compatible with the inherent periodicity of the Fourier transform. At each time step, the field data in the guard cells are reset to zero before each field update. After the update, the field has propagated some distance in the guard cell region, whose size is set so that the amplitude of the field data at the outer edge of the region is negligible. This ensures that the values in the outer cells of the subdomain on which the forward and backward FFTs are performed are essentially zero, verifying periodicity.

The simplest implementation of PSATD with domain decomposition consists in performing one FFT per core on CPU-based computers and one FFT per GPU on GPU-based computers, and its performance is discussed later.

A. Weak scaling studies on Summit with 1 FFT/GPU

Figure 1 shows the results of a set of weak scaling tests with WarpX, with a simulation grid of $N_x \times N_y \times N_z = 768 \times 512 \times 256$ cells per node, with no macroparticles (PSATD solver only), when using 8, 16, or 32 guard cells to surround the computational grid of each message passing interface (MPI) subdomain. The scalings were performed on a large range of nodes on Summit, from 1 to 4096 nodes.

The scalings have a similar trend with regard to the number of nodes, regardless of the number of guard cells, with a significant increase in the runtime from 1 node to 8 nodes, followed by a plateau with a gentle slope upward. The initial increase is due to the topology of the MPI subdomains across the nodes and the exchanges of guard regions' data between nodes, as explained in the Appendix.

The plateau is indicating that the PSATD solver is scaling well up to the 4096 nodes, even for a number of guard cells, that is, as large as 32. The dependency of the runtime with regard to the number of guard cells is roughly linear, indicating that it is dominated by the exchanges of data from the guard regions. More precisely, the guard cell regions at each end of a subdomain have $(N_y + 2N_{guards}) \times (N_z + 2N_{guards}) \times N_{guards}$, $(N_x + 2N_{guards}) \times (N_z + 2N_{guards}) \times N_{guards}$, and $(N_x + 2N_{guards}) \times (N_y + 2N_{guards}) \times N_{guards}$ cells along x, y, and z, respectively. Using $\{N_x, N_y, N_z\} = \{768, 512, 256\}$ and $N_{guards} = 8$, 16, or 32, one finds that the ratio of guard cells is ≈ 2.1 and ≈ 4.8 when going from 8 to 16 and from 8 to 32 guard cells, respectively, which is in fair agreement with the ratios in computing time observed in Fig. 1.

Figure 2 reports on a similar weak scaling without macroparticles and with 8 macroparticles/cell. The simulation grid was downsized to $N_x \times N_y \times N_z = 384 \times 256 \times 128$ cells per node, in order to fit all the grid and macroparticles data on each GPU memory. The scalings were performed from 1 to 128 nodes.

The trend of the scalings with regard to the number of nodes is the same as before, with an initial increase from 1 to 8 nodes, followed by a plateau. Without macroparticles, the runtime is still roughly directly proportional to the number of guard cells when going from

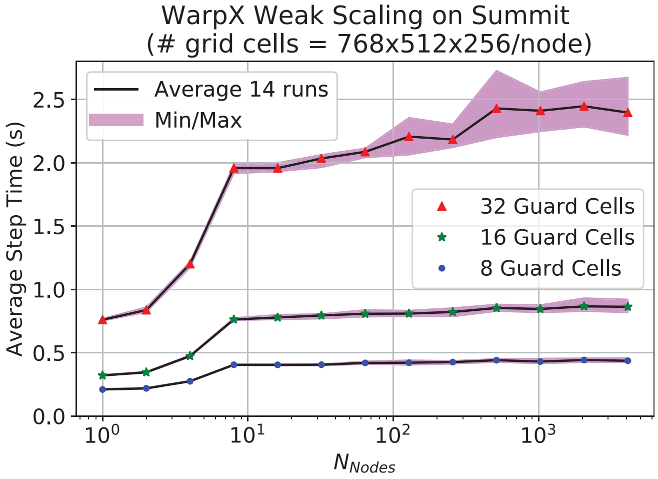

FIG. 1. Average clock time (in seconds) per time step as a function of the number of nodes from a weak parallel scaling study with WarpX, using the PSATD solver with 1FFT/GPU, for 8, 16, or 32 guard cells, without particles (PSATD solver only). Each value is the average of 14 time steps. The range between the maximum and the minimum values is plotted as a filled color background behind each curve. [Associated dataset available at https://doi.org/10.5281/zenodo.4429368] (Ref. 32).

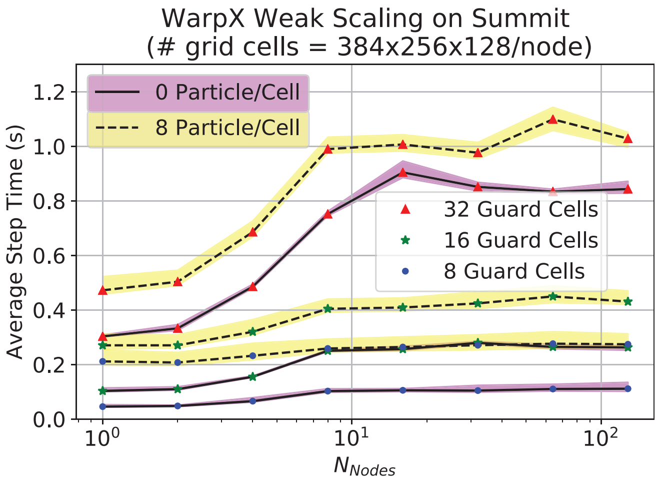

FIG. 2. Average clock time (in seconds) per time step as a function of the number of nodes from a weak parallel scaling study with WarpX, using the PSATD solver with 1FFT/GPU, for 8, 16, or 32 guard cells, without particles or with 8 particle/cell. For the case with particles, a plasma is loaded uniformly in the computational domain (perfect load balancing). Each value is the average of 14 time steps. The range between the maximum and the minimum values is plotted as a filled color background behind each curve. [Associated dataset available at https://doi.org/10.5281/zenodo.4429368] (Ref. 32).

8 to 16 guard cells but is proportionally more costly when going from 16 to 32 guard cells, due to an increased ratio of the number of guard cells vs number of grid cells. Adding particles adds computing and data exchanges that are independent of the number of guard cells (there are no duplications of macroparticles in the guard regions), mitigating the dependency of the runtime with the thickness of the guard cell regions to some extent.

The conclusions of these scaling studies are that (a) the spectral solver is exhibiting very good weak scaling on up to the 4096 nodes on Summit and (b) the runtime cost of the PSATD solver is dominated by the cost of MPI exchanges of the guard regions' data (as opposed to the cost of the FFTs). It is, thus, of paramount importance to identify solutions that can minimize the number of guard cells to use for a given stencil order, reduce the amount of data to be exchanged for a given number of guard cells, or both. Possible solutions are discussed in Sec. III B.

B. Exploration of solutions for possible improvement

1. Minimization of the number of guard cells

The number of guard cells is typically proportional to the selected order of the finite approximation of the spatial derivatives. It is, thus, imperative to balance the order with the cost of MPI exchanges of data from guard regions. With standard finite-difference solvers, the number of guard cells is set to cover the spatial footprint of the stencil at a given order. With PSATD, the analytical time integration spreads the footprint of the stencil to infinity. However, the value of the equivalent finite-difference weights falls below machine precision at a given number of cells from the center of the finite spatial derivation, that is, proportional to the order. A conservative approach consists, thus, in setting the number of guard cells to the number of cells beyond which the weights are below machine precision.[31] A more aggressive approach consists in truncating the stencil by using a smaller number of guard cells, such that the truncation error is below a given threshold,[5,30] e.g., 10^{-6}. The former has been used successfully with our axisymmetrical pseudo-spectral code FBPIC on the modeling of plasma accelerators.[31,33–36] The latter has been used successfully with our code Warp + PICSAR for the modeling of plasma mirrors.[36,37]

A complete study of how to, and to what threshold, truncate the stencil requires many considerations that involve functionalities that were implemented recently or are planned. Such a study will be conducted when all the functionalities are operational and will be summarized in a future publication.

2. Hybrid local-parallel FFTs

A hybrid implementation, where a parallel FFT is performed on a group of CPUs, was explored for the IBM BG-Q Mira and the CRAY XC40 Theta computers at the ALCF.[38] The reader is referred to the publication[38] for the details of the procedure and tests that were performed. For brevity, these are not repeated here and we focus on the main conclusion, which is that the results demonstrated the advantage of the hybrid approach on CPU-based machines when using a large number of guard cells for very high-order stencils. On Theta, the lowest runtimes were obtained when performing parallel FFTs on groups of 1/64/1024 MPI processes when using, respectively, 8/16/32 guard cells. On Mira, the lowest runtimes were obtained when performing parallel FFTs on groups of 16/128/1024/8192 MPI processes when using, respectively, 8/16/32/64 guard cells.

Because the PSATD solver is scaling very well with 1 FFT per GPU, even in the somewhat extreme case of guard regions that are 32 cells thick, a hybrid solution is not absolutely necessary. Preliminary runtime testings of parallel FFTs on one and two Summit nodes (not reported here) did not give results that would lead to an unequivocal conclusion that a hybrid solution will lead to faster runtimes. However, because (i) the implementation of efficient parallel FFTs on Summit is still in progress and may lead to shorter runtime in the future and (ii) there are added benefits in stability of parallel PSATD by performing the FFTs on larger subdomains, we are exploring further the possibility with various parallel FFT packages. Progress and findings will be reported in a future paper.

3. Reduced precision

We recently added options to control the floating-point precision of field and particle types in WarpX. While our integration tests comparing WarpX simulations with analytical models already confirm the expected level of machine precision for physical quantities of interest, large-scale tests and the influence on particle beam properties such as beam emittance are ongoing.

In terms of communication and scaling improvements, reduced precision to single-precision cuts the communicated data size for fields in half and improves computational speed on GPUs. As measured in Fig. 3, for cross-device communication-dominated setups with many guard cells, single precision usage can reduce the average step time by a factor 2×.

This result is intriguing for further investigation for FFT-solvers and more sensitive than our measurements for simulations that use stencil solvers, which are not cross-device communication but

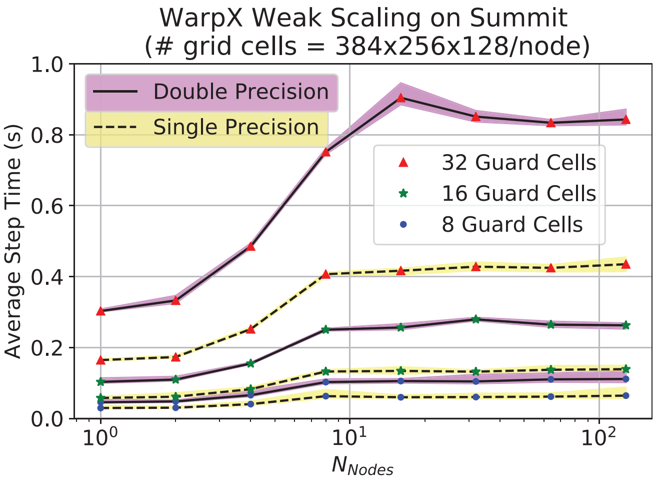

FIG. 3. Average clock time (in seconds) per time step as a function of the number of nodes from a weak parallel scaling study with WarpX, using the PSATD solver with 1FFT/GPU, for 8, 16, or 32 guard cells, in double precision (solid lines) and single precision (dashed lines). Each value is the average of 14 time steps. The range between the maximum and the minimum values is plotted as a filled color background behind each curve. [Associated dataset available at https://doi.org/10.5281/zenodo.4429368] (Ref. 32).

on-device memory dominated, and usually only show a speedup of 1.5–1.6× when switching to single precision. In particular, FDTD kernels do not reach a 2× speedup when switching to single precision,[39] since many auxiliary operations that are executed per GPU kernel, i.e., indexing operations (integer math) and atomic reductions in current depositions, cannot be expected to speed up linearly when reducing floating-point memory transfer bandwidth.

Further studies are under way to determine the effect of single precision on accuracy, especially when using very high-order stencils with the pseudo-spectral Maxwell solver.

IV. SIMULATION OF THREE CONSECUTIVE MULTI-GeV PLASMA ACCELERATOR STAGES

As mentioned in Sec. I, WarpX is being developed to enable first-principle, three-dimensional modeling of chains of tens to hundreds of plasma accelerating stages on exascale supercomputers, for future collider designs. To demonstrate the code's capabilities, we have recently focused on modeling the acceleration of an electron beam with properties similar to those available at the BELLA center at Lawrence Berkeley National Laboratory: with about 150 pC charge and 1 GeV energy. In these simulations, a 3 μm-long electron beam is focused onto a 0.6 μm transverse spot at the entrance of the first electron–proton plasma stage. The plasma electrons and protons of a stage are pre-ionized and quasi-neutral; all particles, including the electron beam, are simulated as macroparticles. In all three stages, the plasma column has a transverse parabolic profile of the form $n_0[1 + 4r^2/(k_p^2 R_c^4)]$, with a radius of curvature $R_c = 40\,\mu$ m and a density on axis $n_0 = 1.7 \times 10^{23}$ cm^{-3}, as a function of the plasma wavenumber k_p. Longitudinally, the plasma ramps up and down linearly over 2 and 0.5 cm, respectively, encompassing a constant density section of 29.5 cm (such that the total length of each stage is 30 cm). In each of the two gaps (both 3 cm long) between the stages, there is a mirror that prevents the laser beam that exits one stage from entering the following stage, as well as a lens (idealized as constant focusing electric field, uniform longitudinally over the length of the lens and with an amplitude that grows linearly in the radial direction) that focuses the beam (which expands when exiting a stage due to its intrinsic transverse velocity spread) at the entrance of the next stage. It is crucial to focus the beam at a radius that is as close as possible to the plasma matching condition at the entrance of the column, so as to reduce betatron oscillations and preserve the beam quality (i.e., transverse size and velocity spread), as required for colliders and other applications. A diagram of the setup is given in Fig. 4.

To speed up the simulation, it was conducted with a Lorentz boosted frame of reference[4] with a relativistic factor of $\gamma_b = 60$. A moving window was also employed to follow the beam as it progresses through the chain of stages. Simulations were performed using the PSATD solver or with an NFD FDTD solver.[21,23] For control of the numerical Cherenkov instability,[40] a specifically tailored nine-points filter[41] was used in the FDTD simulations in the longitudinal direction, while the Galilean PSATD scheme[42,43] was used in the PSATD simulations. For the latter, the order of the stencil for the Maxwell solver was set to 16, using 16 guard cells in the transverse direction and 20 guard cells in the longitudinal direction (the asymmetry in the number of guard cells is related to the asymmetrical nature of the Galilean scheme, which integrates Maxwell's equations in a Galilean frame moving along z). The plasma electrons and protons were initialized uniformly in a 3D Cartesian geometry using a 2 × 2 × 1 pattern along x, y, and z, respectively, for an initial 4 macroparticles per cell for each species. The choice of the pattern was made based on results of a scan varying the initial number of plasma macroparticles per cell along each direction.

Figure 5 depicts the beam propagating along the different plasma stages, from a 3D WarpX simulation using the PSATD solver. The expansion of the beam as it exits the first stage is clearly visible, as well as its refocusing by the (idealized) plasma lens located at one centimeter in front of the second stage. The displacement of the plasma electrons, initially uniformly distributed, caused by the laser's ponderomotive force, is also clearly visible. This is the displacement that creates a pocket of positive charge density (from the underlying, barely mobile ions), giving rise to high-gradient electric fields that can accelerate the electron beam to very high energy over a very short distance.[44,45]

Scans of the longitudinal and transverse resolutions were performed to assess the level of convergence of the beam final properties. The histories of the laboratory frame electron beam moments (mean energy, relative energy spread, averaged slice emittances in X and Y, and beam RMS width in X and Y) are plotted in Fig. 6, while the dependency of the final values with resolution are plotted in Fig. 7, for the resolutions $\Delta x = \Delta y = \Delta z = 5,\ 2.5,\ 1.25$ and $0.625\,\mu$m (the longitudinal cell size Δz is given in the Lorentz boosted frame of the

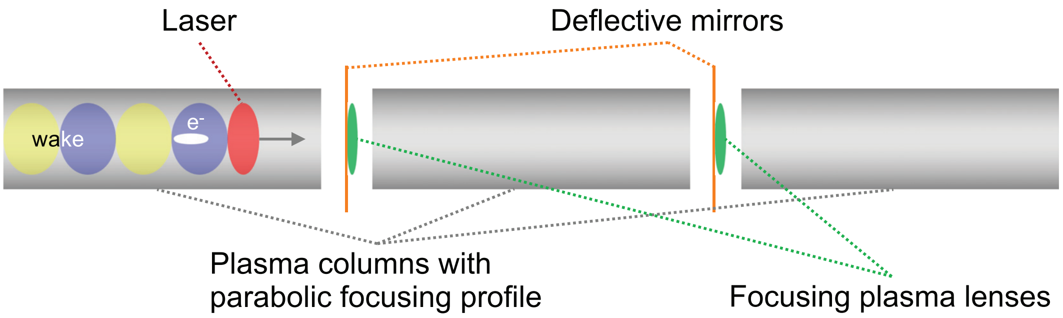

FIG. 4. Diagram of the simulated setup. A laser beam (red) enters in a chain of three plasma columns, creating a wake with oscillating high-gradient electric fields (yellow and blue) that are used to accelerate an electron beam (white) over a very short distance. The plasma columns have a transverse parabolic profile, which provides transverse focusing of the laser that would, otherwise, diffract. Deflective mirrors (orange lines) are inserted between each stage to prevent the laser beam that exits one stage (and has largely depleted its energy) from entering the following stage, as new laser beam is injected in the following stage. Plasma lenses (green) refocus the electron beam between each stage.

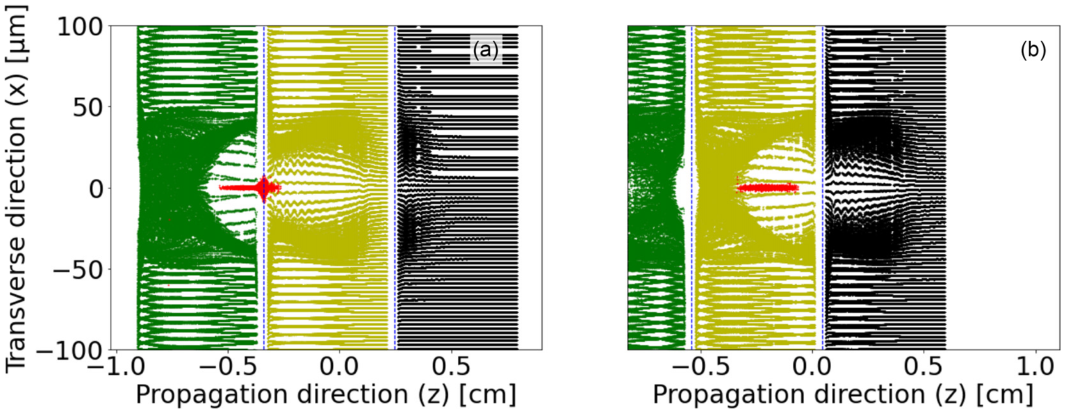

FIG. 5. Snapshots from 3D WarpX simulations as the accelerated electron beam (in red, propagating from left to right): (a) exits the first plasma column and (b) propagates in the second plasma column. The macroparticles from the central slice along Y of the plasma electrons are plotted in green, yellow, and black for stages 1, 2, and 3, respectively. The electron beam expands radially at the exit of each stage and is refocused transversely by a lens (blue dashed line). Note that due to Lorentz dilation/contraction from the use of a Lorentz boosted frame at $\gamma = 60$, the beam appears almost 120 times longer, and the plasma chain appears 60 times shorter, than in the laboratory.

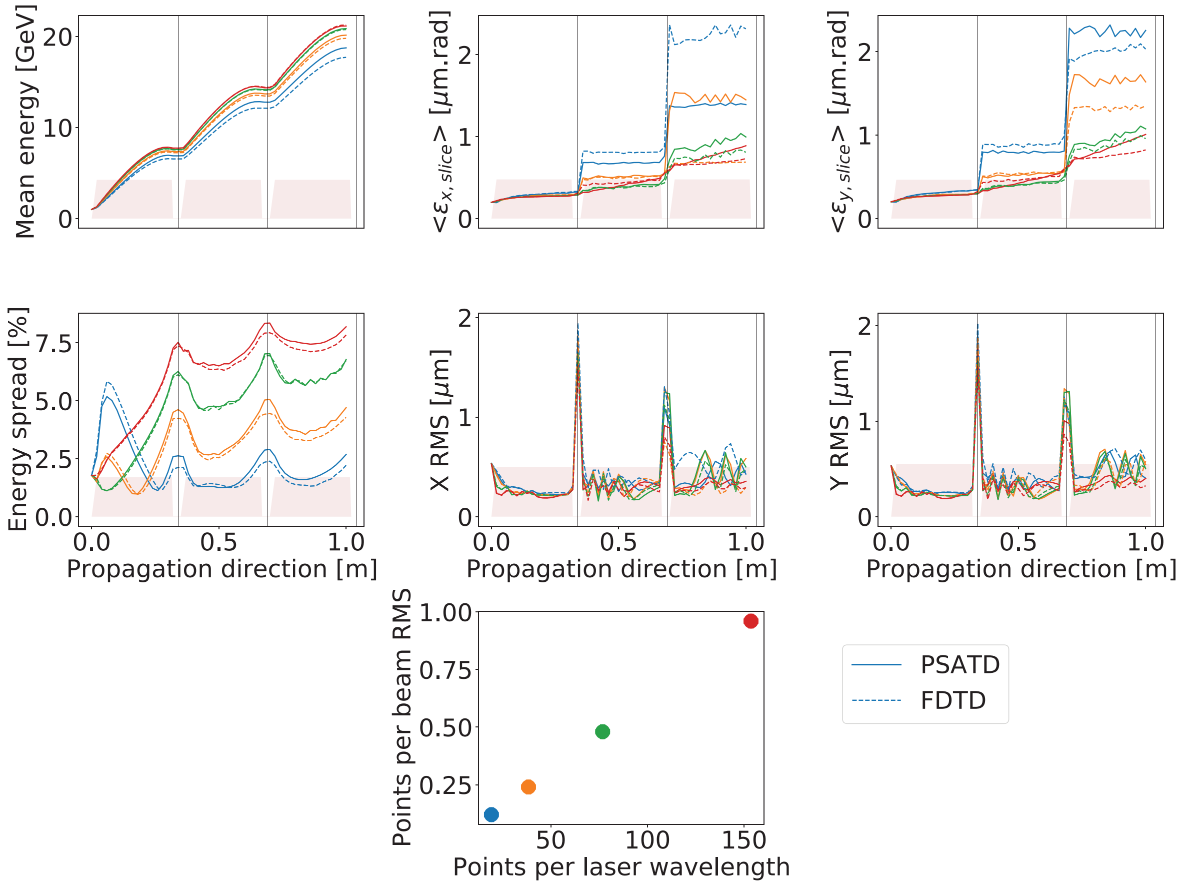

FIG. 6. Evolution of the laboratory frame electron beam moments (mean energy, relative energy spread, averaged slice emittances in X and Y, and beam RMS width in X and Y) as it is accelerated in a chain of three consecutive laser-driven plasma accelerator stages. Results are given from simulations using the PSATD solver (solid lines) or the FDTD solver (dashed lines). Four resolutions were considered (given in the boosted frame of simulation): $\Delta x = \Delta y = \Delta z = 5$ μm (blue), 2.5 μm (yellow), 1.25 μm (green), and 0.625 μm (red). The lines color corresponds to the longitudinal and transverse resolution relative to the laser wavelength and the electron beam initial transverse RMS size, as given in the bottom plot. The longitudinal profile of the plasma density is depicted by the regions filled in light pink at the bottom of the top and middle plots. The solid vertical gray lines mark the location of the two focusing lenses. [Associated dataset available at https://doi.org/10.5281/zenodo.4429368] (Ref. 32).

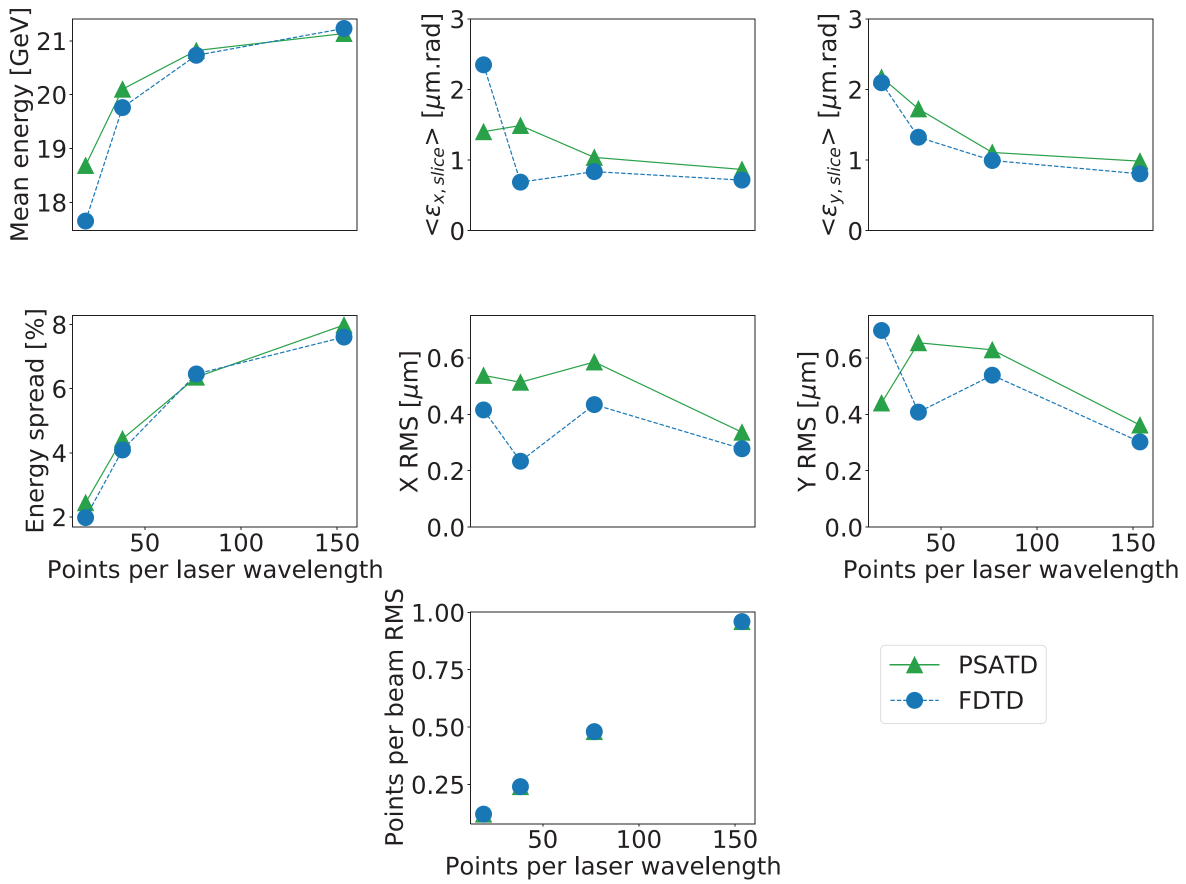

FIG. 7. Final values of the lab frame electron beam moments (mean energy, relative energy spread, averaged slice emittances in X and Y, and beam RMS width in X and Y) at the exit of the 3 stages, as a function of longitudinal and transverse resolution: $\Delta x = \Delta y = \Delta z = 5\,\mu$m, 2.5 μm, 1.25 μm, and 0.625 μm, respectively. [Associated dataset available at https://doi.org/10.5281/zenodo.4429368] (Ref. 32).

simulation). To compute the moments, two categories of macroparticles were excluded: first, all macroparticles beyond a $10\,\mu$m radius, to remove particles that separated from the beam core; second, all macroparticles located beyond three root mean square of the 6D particles position and velocity hyperradius $r = \sqrt{\tilde{x}^2 + \tilde{y}^2 + \tilde{z}^2 + \tilde{u}_x^2 + \tilde{u}_y^2 + \tilde{u}_z^2}$, where $\{x, y, z\}$ and $\{u_x, u_y, u_z\}$ are the position and velocity of the particles, respectively, and $\tilde{a} = (a - \bar{a})/\sigma(a)$, where $\bar{a}$ and $\sigma(a)$ are, respectively, the average and the standard deviation of the quantity a. The particles located outside this hypervolume are likely to be lost further downstream and were, thus, also excluded from the moments calculations. The average slice emittances along X and Y are the average of the emittances of 20 beam slices, weighted by the relative number of macroparticles in each slice.

The average beam energy is increasing monotonically within each stage, with a small deceleration toward the exit of the stages and a short plateau between stages. The predicted average energy from all the simulations (whether using PSATD or FDTD) is nicely converging toward the same results as the resolution is increased. The energy spread history is highly dependent on resolution very early at the entrance of the first stage, resulting in values that are well separated for resolutions varying from 5 μm to 0.625 μm. The amplitude of the separation is nonetheless diminishing with increasing resolution, indicating a trend toward convergence. Despite not reaching the level of convergence that is reached with the average energy, the results and trends agree quite well between the PSATD and FDTD runs.

The evolutions of the average slice emittances along X and Y show sharp increase in the gaps, followed by relatively steady plateaus inside stages, with the notable exception of the highest resolution runs with the PSATD solver for which the emittance growth is more progressive. Looking solely at the dependency of the final emittances vs numerical resolution in Fig. 7, there is nonetheless a trend toward similar converged values in X and Y with the FDTD and PSATD solvers. This is the case also for the RMS transverse beam sizes, which also exhibit similar trends and amplitudes along X and Y at the two highest resolutions with both the FDTD and PSATD solvers. The lower degree of convergence of the beam RMS sizes, compared to other moments, is to be expected, as the beam is experiencing many betatron oscillations in each stage. It is, thus, more difficult to reach convergence for the transverse beam RMS sizes, where both the amplitude and the phase of the oscillations matter, contrarily to the mean energy, energy spread, and emittances, which do not undergo such oscillations.

The simulations were run on OLCF's Summit supercomputer, using from 6 to 1536 GPUs, with the PSATD solver and the FDTD solver runs. In the series of simulations that were performed, the runs with the PSATD solver were between 40% and twice slower than the FDTD ones. While the FDTD solver is more efficient with this particular proof-of-principle series of runs, the PSATD solver offers significant advantages that will put it ahead in various situations, like:

- a higher tolerance to the numerical Cherenkov instability.[46] Indeed, simulations of the same setup with a higher Lorentz boost factor γ_{boost} of 85 (measured to be the relativistic factor associated with the group velocity of the laser in the plasma columns in the laboratory frame, which is optimal for these types of simulations[47]) were not stable with the FDTD solver but stable with the PSATD Galilean solver. The additional speedup[4] of $\approx(85/60)^2 \approx 2$ results in the runs of the PSATD solvers at $\gamma_{boost} = 85$ being as fast or faster than the ones with the FDTD solver at $\gamma_{boost} = 60$ for the setup considered in this paper. The gain can be even higher for other setups that allow for higher γ_{boost}.
- a path for accurate and efficient implementation of mesh refinement. As reported in previous papers,[7,48] mesh refinement in electromagnetic PIC codes depends critically on low numerical dispersion, which calls for high-order Maxwell solvers such as PSATD.
- the PSATD solver itself is not subject to a Courant condition, opening the path to explicit PIC solvers that can enable larger time steps in cases where, e.g., $c\Delta t = \Delta x \ll \Delta z$ with $v_x \ll v_z$, where v_x and v_z are, respectively, the maximum velocities of macroparticles along x and z. Such kinds of novel solvers are being explored.

These series of simulations have cross-validated the PSATD- and FDTD-based PIC solvers that are implemented in WarpX, on the modeling of chains of three consecutive multi-GeV plasma accelerator stages, a prerequisite for the modeling of tens to hundreds of plasma stages that will be needed for possible future plasma-based collider designs. Convergence was verified for the non-oscillating accelerated electron beam quantities such as its beam energy, energy spread, and transverse emittances. Convergence of quantities like the beam horizontal and vertical transverse sizes, which oscillate at the betatron frequency, is more difficult to reach due to the additional difficulty of converging on the phase of a rapidly oscillating quantity. In that regard, the results indicate that fairly high resolutions are needed to approach convergence, demonstrating the need for scalable, fast, and accurate simulations tools, and for further study and development of cutting-edge numerical methods that can lower the computational cost, such as mesh refinement.

V. CONCLUSION

The electromagnetic PIC code WarpX has now been fully transitioned from an original mix of Fortran and C++ source code to a fully modern C++14 application. In the process, it has been ported to GPUs (with current support for NVidia GPUs), while preserving its ability to run efficiently on CPUs, using a single-source programing approach via abstractions that are provided through the AMReX library in the form of generic interfaces to parallel-for and reduction primitives.[16] This approach will enable support of other GPU architectures (e.g., from Intel and AMD) with little to no change to numerical algorithms implemented in WarpX. The implementation includes both finite-difference and pseudo-spectral (FFT based) Maxwell solvers. Domain decomposition is used with both types of solvers, and the overall efficiency depends critically on the number of guard cells, that is, used along each dimension, as exchange of data between neighboring subdomains dominate the runtime. The weak scaling of the pseudo-spectral solver has been studied, demonstrating very good weak scaling with only around 25% degradation from 8 to 4096 nodes (with 6 GPUs/node). Finally, the accuracy and convergence of the new implementation were verified on 3D simulations of laser-driven plasma accelerators with both the finite-difference and pseudo-spectral Maxwell solvers. Ongoing work is focused on further optimizing the code (including minimizing the number of guard cells when using the pseudo-spectral solver), improving dynamic load balancing and finalizing the mesh refinement capabilities for production runs.

ACKNOWLEDGMENTS

This research was supported by the Exascale Computing Project (No. 17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative. This work was performed in part under the auspices of the U.S. Department of Energy by Lawrence Berkeley National Laboratory under Contract No. DE-AC02–05CH11231, SLAC National Accelerator Laboratory under Contract No. AC02–76SF00515, and Lawrence Livermore National Laboratory under Contract No. DE-AC52–07NA27344.

This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract No. DE-AC05–00OR22725.

APPENDIX: DEPENDENCY OF WEAK SCALING WITH MPI TOPOLOGY

All the scalings reported in Figs. 1–3 exhibit a significant slowdown when going from 1 to 8 nodes and are relatively flat beyond 8 nodes. This can be explained by examining the relative amount of guard regions' data that is exchanged between nodes for the various runs.

The layout of the MPI subdomains on the various nodes is shown in Figs. 8 and 9 for the runs with 1, 2, 4, and 8 nodes. In all simulations, there are 1 MPI subdomain per GPU and, thus, 6 MPI subdomains per node. Counting the relative amount of guard regions' data that is exchanged between nodes involves the addition of MPI subdomains sides that are adjacent across nodes (to first approximation, the amount of data exchanged from the edges and corners of the guard regions is negligible). Note that the simulations involve periodic boundary conditions in all directions, adding to the amount of data being communicated. For the simulations with 2 nodes, the number of off-node neighbors per node is 8. For 4 nodes, the maximum number of off-node neighbors per node is 14, and it is 26 for 8 nodes. The maximum number of off-node neighbors per node was also computed for simulations with 16 nodes or more and remained at 26 for all.

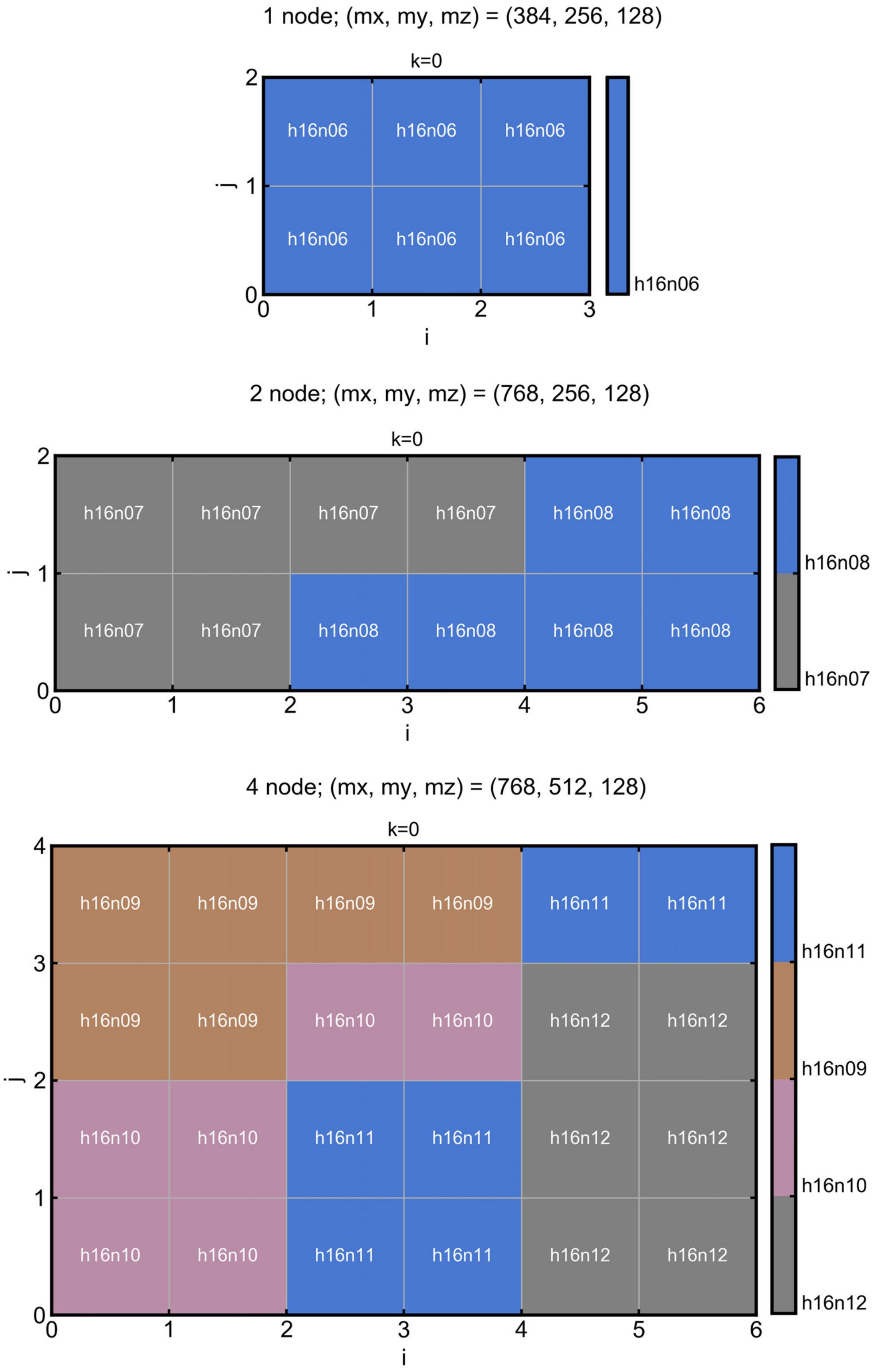

FIG. 8. Layout of MPI subdomains on nodes for simulations using: (top) 1 node, (middle) 2 nodes, and (bottom) 4 nodes. The indices i, j, and k correspond to the positions in the 3D simulation domain along x, y, and z, respectively. Each MPI subdomain is colored according to the node that it belongs to.

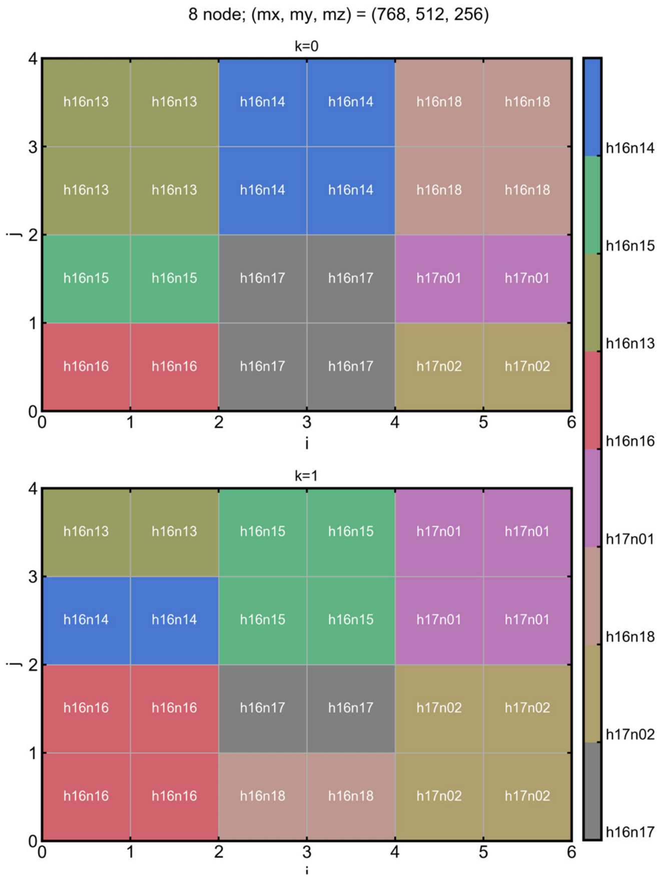
8 node; (mx, my, mz) = (768, 512, 256)
k=0
h16n13 h16n13 h16n14 h16n14 h16n18 h16n18
h16n13 h16n13 h16n14 h16n14 h16n18 h16n18
h16n15 h16n15 h16n17 h16n17 h17n01 h17n01
h16n16 h16n16 h16n17 h16n17 h17n02 h17n02
k=1
h16n13 h16n13 h16n15 h16n15 h17n01 h17n01
h16n14 h16n14 h16n15 h16n15 h17n01 h17n01
h16n16 h16n16 h16n17 h16n17 h17n02 h17n02
h16n16 h16n16 h16n18 h16n18 h17n02 h17n02
i
j
0 1 2 3 4 5 6
0 1 2 3 4
h16n14
h16n15
h16n13
h16n16
h17n01
h16n18
h17n02
h16n17

FIG. 9. Layout of MPI subdomains on nodes for simulations using 8 nodes. The indices i, j, and k correspond to the positions in the 3D simulation domain along x, y, and z, respectively. Each MPI subdomain is colored according to the node that it belongs to.

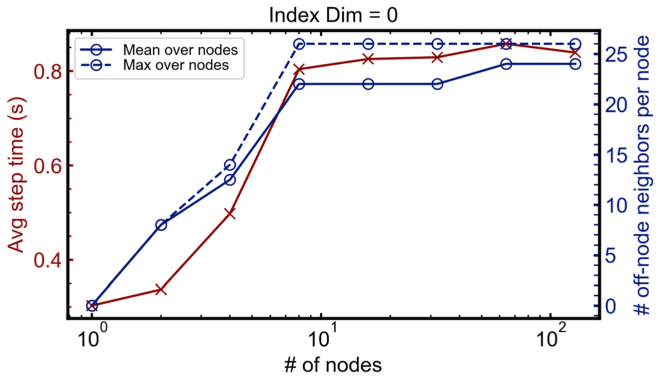

FIG. 10. Average runtime per time step (left axis; solid red line with crosses) and number of off-node neighbors per node—average and maximum (right axis; solid and dashed blue lines with circles) for runs on 1 to 128 nodes.

The average and maximum numbers of off-node neighbors per node (neglecting edges and corners and taking periodicity into account, as explained above) are plotted in Fig. 10 and contrasted to the average runtime per time step. There is a clear correlation between the runtime and relative amount of guard regions' data exchanged between nodes, explaining the trend of the weak scaling from 1 to 8 nodes and beyond.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] ALEGRO Collaboration, "Towards an advanced linear international collider," arXiv:1901.10370 (2019).

[2] See https://github.com/ECP-WarpX/WarpX for "Warpx."

[3] See https://www.exascaleproject.org/ for "Home page-exascale computing project."

[4] J.-L. Vay, "Noninvariance of space- and time-scale ranges under a lorentz transformation and the implications for the study of relativistic interactions," Phys. Rev. Lett. **98**, 130405/1–4 (2007).

[5] H. Vincenti and J.-L. Vay, "Ultrahigh-order Maxwell solver with extreme scalability for electromagnetic PIC simulations of plasmas," Comput. Phys. Commun. **228**, 22 (2018).

[6] J.-L. Vay, A. Almgren, J. Bell, L. Ge, D. Grote, M. Hogan, O. Kononenko, R. Lehe, A. Myers, C. Ng, J. Park, R. Ryne, O. Shapoval, M. Thévenet, and W. Zhang, "Warp-X: A new exascale computing platform for beam-plasma simulations," Nucl. Instrum. Methods Phys. Res., Sect. A **909**, 476–479 (2018).

[7] J. L. Vay, A. Almgren, J. Bell, R. Lehe, A. Myers, J. Park, O. Shapoval, M. Thevenet, W. Zhang, D. P. Grote, M. Hogan, L. Ge, and C. Ng, "Toward plasma wakefield simulations at exascale," in *Proceedings of the 2018 IEEE Advanced Accelerator Concepts Workshop (ACC)* (Institute of Electrical and Electronics Engineers, Inc., 2019).

[8] E. Zenker, R. Widera, A. Huebl, G. Juckeland, A. Knüpfer, W. E. Nagel, and M. Bussmann, "Performance-portable many-core plasma simulations: Porting PIConGPU to open power and beyond," in *High Performance Computing*, edited by M. Taufer, B. Mohr, and J. M. Kunkel (Springer International Publishing, Cham, 2016), pp. 293–301.

[9] W. Zhang, A. Almgren, V. Beckner, J. Bell, J. Blaschke, C. Chan, M. Day, B. Friesen, K. Gott, D. Graves, M. P. Katz, A. Myers, T. Nguyen, A. Nonaka, M. Rosso, S. Williams, and M. Zingale, "AMReX: A framework for block-structured adaptive mesh refinement," J. Open Source Software **4**, 1370 (2019).

[10] H. C. Edwards, C. R. Trott, and D. Sunderland, "Kokkos: Enabling manycore performance portability through polymorphic memory access patterns," J. Parallel Distrib. Comput. **74**, 3202–3216 (2014).

[11] D. A. Beckingsale, J. Burmark, R. Hornung, H. Jones, W. Killian, A. J. Kunen, O. Pearce, P. Robinson, B. S. Ryujin, and T. R. Scogland, "RAJA: Portable performance for large-scale scientific applications," in Proceedings of the IEEE/ACM International Workshop on Performance, Portability and Productivity in HPC (P3HPC), 2019, pp. 71–81.

[12] Khronos OpenCL Working Group SYCL Subgroup, "SYCL Specification," (SYCL, 2019), see https://www.khronos.org/registry/SYCL/specs/sycl-1.2.1.pdf.

[13] J. L. Vay, A. Almgren, L. D. Amorim, J. Bell, L. Ge, K. Gott, D. P. Grote, M. Hogan, A. Huebl, R. Jambunathan, R. Lehe, A. Myers, C. Ng, J. Park, M. Rowan, O. Shapoval, M. Thévenet, W. Zhang, Y. Zhao, and E. Zoni, "Toward the modeling of chains of plasma accelerator stages with WarpX," J. Phys.: Conf. Ser. **1596**, 012059 (2020).

[14] Advanced Micro Devices, Inc. (AMD), *HIP Documentation* (Advanced Micro Devices, Inc., 2019), see https://rocmdocs.amd.com/en/latest/Programming_Guides/Programming-Guides.html.

[15] M. Bussmann, H. Burau, T. E. Cowan, A. Debus, A. Huebl, G. Juckeland, T. Kluge, W. E. Nagel, R. Pausch, F. Schmitt, U. Schramm, J. Schuchart, and R. Widera, "Radiative signatures of the relativistic kelvin-helmholtz instability," in *Proceedings of the International Conference on High Performance Computing,*

Networking, Storage and Analysis (SC) (Association for Computing Machinery, New York, NY, 2013).

[16] A. Myers, A. Almgren, D. Amorim, J. Bell, L. Fedeli, L. Ge, K. Gott, D. Grote, M. Hogan, A. Huebl, R. Jumbunathan, R. Lehe, C. Ng, M. Rowan, O. Shapoval, M. Thévenet, J.-L. Vay, H. Vincenti, E. Yang, N. Zain, W. Zhang, Y. Zhao, and E. Zoni, "Porting WarpX to GPU-accelerated platforms," arXiv:2101.12149 (2021).

[17] S. S. Vazhkudai, B. R. de Supinski, A. S. Bland, A. Geist, J. Sexton, J. Kahle, C. J. Zimmer, S. Atchley, S. Oral, D. E. Maxwell, V. G. V. Larrea, A. Bertsch, R. Goldstone, W. Joubert, C. Chambreau, D. Appelhans, R. Blackmore, B. Casses, G. Chochia, G. Davison, M. A. Ezell, T. Gooding, E. Gonsiorowski, L. Grinberg, B. Hanson, B. Hartner, I. Karlin, M. L. Leininger, D. Leverman, C. Marroquin, A. Moody, M. Ohmacht, R. Pankajakshan, F. Pizzano, J. H. Rogers, B. Rosenburg, D. Schmidt, M. Shankar, F. Wang, P. Watson, B. Walkup, L. D. Weems, and J. Yin, "The design, deployment, and evaluation of the coral pre-exascale systems," in Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, 2018, pp. 661–672.

[18] C. K. Birdsall and A. B. Langdon, *Plasma Physics via Computer Simulation* (Adam-Hilger, 1991), p. Xxvi+479 Pp.

[19] K. Yee, "Numerical solution of initial boundary value problems involving Maxwells equations in isotropic media," IEEE Trans. Antennas Propag. **Ap14**, 302–307 (1966).

[20] A. Pukhov, "Three-dimensional electromagnetic relativistic particle-in-cell code VLPL (Virtual Laser Plasma Lab)," J. Plasma Phys. **61**, 425–433 (1999).

[21] M. Karkkainen, E. Gjonaj, T. Lau, and T. Weiland, "Low-dispersionwake field calculation tools," in Proceedings of the International Computational Accelerator Physics Conference, Chamonix, France, 2006, pp. 35–40.

[22] J. L. Vay, C. G. R. Geddes, E. Cormier-Michel, and D. P. Grote, "Numerical methods for instability mitigation in the modeling of laser wakefield accelerators in a lorentz-boosted frame," J. Comput. Phys. **230**, 5908–5929 (2011).

[23] B. M. Cowan, D. L. Bruhwiler, J. R. Cary, E. Cormier-Michel, and C. G. R. Geddes, "Generalized algorithm for control of numerical dispersion in explicit time-domain electromagnetic simulations," Phys. Rev. Spec. Top.-Accel. Beams **16**, 041303 (2013).

[24] R. Lehe, A. Lifschitz, C. Thaury, V. Malka, and X. Davoine, "Numerical growth of emittance in simulations of laser-wakefield acceleration," Phys. Rev. Spec. Top. -Accel. Beams **16**, 021301 (2013).

[25] J. Cole, "A high-accuracy realization of the yee algorithm using non-standard finite differences," IEEE Trans. Microwave Theory Tech. **45**, 991–996 (1997).

[26] J. Cole, "High-accuracy yee algorithm based on nonstandard finite differences: New developments and verifications," IEEE Trans. Antennas Propag. **50**, 1185–1191 (2002).

[27] B. Gustafsson, H.-O. Kreiss, and J. Oliger, *Time Dependent Problems and Difference Methods* (Wiley, 1995).

[28] I. Haber, R. Lee, H. Klein, and J. P. Boris, "Advances in electromagnetic simulation techniques," in Proceedings of the Sixth Conference on Numerical Simulation Plasmas, Berkeley, CA, 1973, pp. 46–48.

[29] J. L. Vay, I. Haber, and B. B. Godfrey, "A domain decomposition method for pseudo-spectral electromagnetic simulations of plasmas," J. Comput. Phys. **243**, 260–268 (2013).

[30] H. Vincenti and J.-L. Vay, "Detailed analysis of the effects of stencil spatial variations with arbitrary high-order finite-difference Maxwell solver," Comput. Phys. Commun. **200**, 147–167 (2016).

[31] S. Jalas, I. Dornmair, R. Lehe, H. Vincenti, J.-L. Vay, M. Kirchen, and A. R. Maier, "Accurate modeling of plasma acceleration with arbitrary order pseudo-spectral particle-in-cell methods," Phys. Plasmas **24**, 033115 (2017).

[32] J.-L. Vay (2021). "Modeling of a chain of three plasma accelerator stages with the warpx electromagnetic pic code on gpus," Zenodo. https://doi.org/10.5281/zenodo.4429368.

[33] V. Kubytskyi, E. Baynar, C. Bruni, K. Cassou, V. Chaumat, N. Delerue, J. Demailly, D. Douillet, D. Garzella, O. Guilbaud, S. Jenzer, S. Kazamias, B. Lucas, G. Maynard, O. Neveu, M. Pittman, R. Prazeres, H. Purwar, D. Ros, and K. Wang, "Laser-plasma acceleration modeling approach in the case of ESCULAP project" in Proceedings of the 10th International Particle Accelerator Conference (IPAC), Melbourne, Australia, 19–24 May 2019, p. THPGW059.

[34] M Kozlova, I Andriyash, J. Gautier, S. Sebban, S. Smartsev, N. Jourdain, U. Chulagain, Y. Azamoum, A. Tafzi, J.-P. Goddet, K. Oubrerie, C. Thaury, A. Rousse, and K. Ta Phuoc, "Hard x rays from laser-wakefield accelerators in density tailored plasmas," Phys. Rev. X **10**, 011061 (2020).

[35] S. R. Yoffe, R. Lehe, B. Ersfeld, E. Brunetti, G. Vieux, A. Noble, B. Eliasson, M. S. Hur, J.-L. Vay, and D. A. Jaroszynski, "Particle-in-cell simulation of plasma-based amplification using a moving window," Phys. Rev. Res. **2**, 013227 (2020).

[36] G. Blaclard, H. Vincenti, R. Lehe, and J. Vay, "Pseudospectral Maxwell solvers for an accurate modeling of Doppler harmonic generation on plasma mirrors with particle-in-cell codes," Phys. Rev. E **96**, 033305 (2017).

[37] A. Leblanc, S. Monchocé, H. Vincenti, S. Kahaly, J.-L. Vay, and F. Quéré, "Spatial properties of high-order harmonic beams from plasma mirrors: A ptychographic study," Phys. Rev. Lett. **119**, 155001 (2017).

[38] H. Kallala, J. L. Vay, and H. Vincenti, "A generalized massively parallel ultra-high order FFT-based Maxwell solver," Comput. Phys. Commun. **244**, 25–34 (2019).

[39] See https://github.com/ComputationalRadiationPhysics/picongpu/issues/2815 for Compare also to PIConGPU 0.4.2,[15] on P100 GPUs that shows the same behavior.

[40] B. Godfrey, "Numerical Cherenkov instabilities in electromagnetic particle codes," J. Comput. Phys. **15**, 504–521 (1974).

[41] B. B. Godfrey and J.-L. Vay, "Suppressing the numerical Cherenkov instability in FDTD PIC codes," J. Comput. Phys. **267**, 1–6 (2014).

[42] R. Lehe, M. Kirchen, B. B. Godfrey, A. R. Maier, and J.-L. Vay, "Elimination of numerical cherenkov instability in flowing-plasma particle-in-cell simulations by using galilean coordinates," Phys. Rev. E **94**, 053305 (2016).

[43] M. Kirchen, R. Lehe, B. B. Godfrey, I. Dornmair, S. Jalas, K. Peters, J.-L. Vay, and A. R. Maier, "Stable discrete representation of relativistically drifting plasmas," Phys. Plasmas **23**, 100704 (2016).

[44] T. Tajima and J. Dawson, "Laser electron-accelerator," Phys. Rev. Lett. **43**, 267–270 (1979).

[45] E. Esarey, C. B. Schroeder, and W. P. Leemans, "Physics of laser-driven plasma-based electron accelerators," Rev. Mod. Phys. **81**, 1229–1285 (2009).

[46] B. B. Godfrey, J. L. Vay, and I. Haber, "Numerical stability analysis of the pseudo-spectral analytical time-domain PIC algorithm," J. Comput. Phys. **258**, 689–704 (2014).

[47] J. L. Vay, C. G. R. Geddes, E. Esarey, C. B. Schroeder, W. P. Leemans, E. Cormier-Michel, and D. P. Grote, "Modeling of 10 GeV-1 TeV laser-plasma accelerators using lorentz boosted simulations," Phys. Plasmas **18**, 123103 (2011).

[48] J.-L. Vay, J.-C. Adam, and A. Heron, "Asymmetric Pml for the absorption of waves. application to mesh refinement in electromagnetic particle-in-cell plasma simulations," Comput. Phys. Commun. **164**, 171–177 (2004).