Abstract

LCODE is a freely-distributed quasistatic 2D3V code for simulating plasma wakefield acceleration, mainly specialized at resource-efficient studies of long-term propagation of ultrarelativistic particle beams in plasmas. The beam is modeled with fully relativistic macro-particles in a simulation window copropagating with the light velocity; the plasma can be simulated with either kinetic or fluid model. Several techniques are used to obtain exceptional numerical stability and precision while maintaining high resource efficiency, enabling LCODE to simulate the evolution of long particle beams over long propagation distances even on a laptop. A recent upgrade enabled LCODE to perform the calculations in parallel. A pipeline of several LCODE processes communicating via MPI (Message-Passing Interface) is capable of executing multiple consecutive time steps of the simulation in a single pass. This approach can speed up the calculations by hundreds of times.

Keywords: Plasma wakefield acceleration, Parallel computations, Message-Passing Interface

1. Introduction

Plasma wakefield acceleration with a particle driver (PWFA) is a novel method of achieving high gradient acceleration of charged particles. The idea boils down to using plasma as a medium to transfer the energy of the driver — a high-energy charged particle beam — to the witness — charged particles that follow the driver. Though the concept dates back to 1985 [1] and its feasibility has been well-proven since that time, experimentally PWFA is still in its infancy, treated mostly as a phenomenon in need for a more detailed investigation, not as a mature method ready to serve immediate high energy physics’ needs.

The current flagship PWFA projects are the already-milestone-setting FACET at SLAC [2] — the one with the highest energy electron and positron beams — and the soon-to-be-launched AWAKE at CERN [3-5] — the first one to use a proton beam as a driver.

Studying PWFA relies heavily on numerical simulations, and calculating the evolution of a macroscopic amount of particles with a reasonable amount of computational resources is a great feat of its own. This raises the need for specialized programs geared towards performance of PWFA simulations.

A significant performance gain may be achieved by using boosted frame for the calculations [6-8]. Another computationally-effective approach is quasistatic approximation [9-13], which allows decoupling the beam simulation time step from the plasma simulation time step at the cost of limited applicability. For example, particle trapping by the plasma wave, sharp longitudinal plasma density gradients or betatron radiation of the beams cannot be self-consistently simulated by quasistatic codes.

2. LCODE

LCODE is a freely-distributed code for simulations of particle beam-driven plasma wakefield acceleration [14-17]. Its main characteristic features are:

- 2D3V plane or axisymmetric geometry,
- co-moving simulation window that moves with the light velocity,
- quasistatic approximation,
- fully kinetic, fully relativistic beam model,
- optional fluid plasma solver,
- field-based kinetic plasma solver,
- suppression of small-scale plasma density noise,
- automatic substepping,
- extensive in-built diagnostics.

Automatic substepping — decreasing simulation step sizes to match the target precision — is implemented in both beam (for low energy particles) and plasma solvers (for fine field structure areas).

Plasma response calculations are done in terms of electric and magnetic fields, rather than potentials, allowing...
to simulate non-trivial configurations, like transversely inhomogeneous, non-uniformly heated, non-neutral plasmas or plasmas with mobile ions of several different sorts.

This unique set of features allows to perform multidimensional parameter scans, consisting of hundreds and thousands full-scale simulation runs [18, 19] in a reasonable amount of time. For example, a single full-scale AWAKE plasma cell [20] simulation takes less than a hundred CPU hours.

3. Quasistatic algorithm

Computationally-wise, the particle beam state and evolution is central to the LCODE operation. LCODE divides the simulation window into layers, processed from head to tail (Figure 1). Ultrarelativistic approximation implies that in the simulation window, which moves with the speed of light, the information can propagate only towards the tail of the beam, so the evolution of each layer depends only on the previous ones. Internally the beam is stored as a collection of the particles, ordered by the layer location. For every layer, the plasma response to the particle beam is calculated with iterative algorithms, then the evolution of the beam is calculated: each particle’s coordinate and momentum get updated in accordance to the plasma response. The procedure is repeated for each layer and for each time step. Slow beam particles can cross the layer borders and thus participate in several layer calculations during one time step. From the technical point of view, at each time step LCODE calculates \((k + 1)\)th state of the beam from the \(k\)th state, layer by layer, head to tail.

4. Parallel operation concept

The algorithm described above is, at its heart, processing a large stream of beam data and calculating an updated version of the beam in a single pass along the simulation window. Thus it is naturally possible to reimplement it as a stateful stream-oriented transformation.

Here ‘stateful’ refers to the fact that the layer calculation still depends heavily on the field and particle information carried over from the previous layers. Mapping this scheme to, e.g., the actor model will introduce excessive complexity because this information would have to be transferred between different processes frequently. On the other hand, pinning a dedicated process to a single time step allows the solvers to simply preserve a part of their state and carry it over from the previous layer, thus avoiding interprocess transfer costs. The only notable downside of such pinning is the need to adhere to a comparably strict data flow topology. For example, as each process possesses unique information about its time step, the number of processes cannot be easily reduced mid-calculation.

‘Stream-oriented’ means that the program consumes and outputs portions of the information regularly and in small portions: while it calculates one time step of the evolution of the beam in a single pass, it outputs the finished layers of the new generation immediately after the calculation and consumes the previous beam generation data as and when necessary. An output stream of data can be immediately processed by another instance of this algorithm as soon as at least one layer of the preceding time step is completed (Figure 2). Extending this pipeline to \(N\) instances allows to calculate \((k + N)\)th state from the \(k\)th one in a single pass.

A parallel computation, designed as a pipeline of simultaneously executing stream transformations, has additional benefits in terms of process interoperation. Most data exchange and synchronization between processes can be reduced to read/write primitives. Moreover, a streaming pipeline does not require shared memory, and is readily scalable to span across multiple hosts.

5. Parallel upgrade implementation

A parallel version of LCODE has been developed recently using the pipeline scheme described above.

This change required to reconsider how simulations, diagnostics and beam storage are implemented within LCODE. Previously the beam information was stored in RAM and was indeed available in its entirety for the diagnostics subsystem. In order to facilitate the implementation of the suggested parallel pipeline upgrade without resorting to shared memory, LCODE was revised to perform all aspects of the simulation, diagnostics and beam storage in a single pass. This ‘stream-oriented’ LCODE is usable as a building block for the suggested parallel processing pipeline; the only thing missing is the data flow between the blocks. The interprocess communication within
the pipeline is implemented using MPI (Message-Passing Interface).

An interesting caveat worth sharing: if the beam data consists only of individual particles’ information, the pipeline will stall upon reaching a long region of no beam particles. A process cannot advance to the next layer until it is sure that it has received all of its particles. In the absence of an ‘end-of-the-layer’ message, this can only happen when a particle from the other side of the region is received. As a result, all processes have to pause at the edge of the empty region and traverse it one-by-one. This effect can be mitigated by exchanging empty layer information between the processes.

6. Speedup evaluation

The performance of the parallel version was evaluated on a test task of simulating the AWAKE experiment with a doubled plasma cell length (to increase the amount of computations required) and otherwise reference parameters [20]. The obtained execution times were analyzed in accordance with the Amdahl’s model [22], which classifies the calculations as either strictly parallel (speeding up by $N$ times with $N$ processes) or strictly serial (taking the same time to complete for any number of processes). Results for executions with different numbers of CPU cores (Figure 3) suggest a scalability level corresponding to the fraction of strictly serial execution (β) in the order of 1% or less in Amdahl’s model. To put it simply, using $N$ processes makes more than 99% of the calculations $N$ times faster. The maximum obtained speedup was 177.6 times, which was obtained by completing a 5 days 17 hours job in 46 minutes 60 seconds.

7. Summary

A parallel version of LCODE was implemented as a pipeline of processes, which calculates consecutive time steps of the beam evolution. This parallel version is freely available on-demand. Performance evaluation suggests a nearly-linear speedup for up to hundreds of processes. Parallel upgrade allows to obtain simulation results in hours instead of weeks.

8. Acknowledgements

The authors would like to thank A. V. Petrenko for his help in preparing this article.

This work is supported by The Russian Science Foundation (grant No. 14-12-00043). Performance evaluation was done at Novosibirsk State University Supercomputer Center (NUSC).

References

[1] P. Chen, J. Dawson, R. W. Huff, T. Katsouleas, Acceleration of electrons by the interaction of a bunched electron beam with a plasma, Physical review letters 54 (7) (1985) 693.
[2] M. Litos, E. Adli, W. An, C. Clarke, C. Clayton, S. Corde, J. Delahaye, R. England, A. Fisher, J. Frederico, et al., High-efficiency acceleration of an electron beam in a plasma wakefield accelerator, Nature 515 (7525) (2014) 92–95.
[3] R. Assmann, R. Bingham, T. Bohl, C. Bracco, B. Buttenschön, A. Buttersworth, A. Caldwell, S. Chattopadhyay, S. Cipiccia, E. Feldbauer, et al., Proton-driven plasma wakefield acceleration: a path to the future of high-energy particle physics, Plasma Physics and Controlled Fusion 56 (8) (2014) 084013.
[4] E. Gschwendtner, et al., Submitted to Nuclear Instr. Methods A.
[5] A. Caldwell, et al., Submitted to Nuclear Instr. Methods A.
[6] S. F. Martins, R. Fonseca, W. Lu, W. Mori, L. Silva, Exploring laser-wakefield-accelerator regimes for near-term lasers using particle-in-cell simulation in lorentz-boosted frames, Nature Physics 6 (4) (2010) 311–316.
[7] S. Martins, R. Fonseca, J. Vieira, L. Silva, W. Lu, W. Mori, Modeling laser wakefield accelerator experiments with ultrafast particle-in-cell simulations in boosted framesa), Physics of Plasmas 17 (5) (2010) 056705.
[8] J.-L. Vay, C. Geddes, E. Esarey, C. Schroeder, W. Leemans, E. Cormier-Michel, D. Grote, Modeling of 10 gev-1 tev laser-plasma accelerators using lorentz boosted simulations, Physics of Plasmas 18 (12) (2011) 123103.
[9] P. Mora, T. M. Antonsen Jr, Kinetic modeling of intense, short laser pulses propagating in tenuous plasmas, Physics of Plasmas 4 (1) (1997) 217–229.
[10] S. Morshed, T. Antonsen, J. Palastro, Efficient simulation of electron trapping in laser and plasma wakefield acceleration, Physics of Plasmas 17 (6) (2010) 063106.
[11] N. Jain, J. Palastro, T. Antonsen Jr, W. B. Mori, W. An, Plasma wakefield acceleration studies using the quasi-static code wake, Physics of Plasmas 22 (2) (2015) 023103.
C. Huang, V. K. Decyk, C. Ren, M. Zhou, W. Lu, W. B. Mori, J. H. Cooley, T. Antonsen, T. Katsouleas, Quickpic: A highly efficient particle-in-cell code for modeling wakefield acceleration in plasmas, Journal of Computational Physics 217 (2) (2006) 658–679.

T. Mehrling, C. Benedetti, C. Schroeder, J. Osterhoff, Hipace: a quasi-static particle-in-cell code, Plasma physics and controlled fusion 56 (8) (2014) 084012.

K. Lotov, Simulation of ultrarelativistic beam dynamics in plasma wake-field accelerator, Physics of Plasmas 5 (3) (1998) 785–791.

K. Lotov, Fine wakefield structure in the blowout regime of plasma wakefield accelerators, Physical Review Special Topics-Accelerators and Beams 6 (6) (2003) 061301.

K. Lotov, A. Sosedkin, E. Mesyats, Simulation of self-modulating particle beams in plasma wakefield accelerators, Proceedings of IPAC2013 (Shanghai, China) (2013) 1238–1240.

K. Lotov, Controlled self-modulation of high energy beams in a plasma, Physics of Plasmas (1994-present) 18 (2) (2011) 024501.

K. Lotov, V. Minakov, A. Sosedkin, Parameter sensitivity of plasma wakefields driven by self-modulating proton beams, Physics of Plasmas 21 (8) (2014) 085107.

K. Lotov, A. Sosedkin, A. Petrenko, L. Amorim, J. Vieira, R. Fonseca, L. Silva, E. Gschwendtner, P. Muggli, Electron trapping and acceleration by the plasma wakefield of a self-modulating proton beam, Physics of Plasmas 21 (12) (2014) 123116.

A. H. Karp, H. P. Flatt, Measuring parallel processor performance, Commun. ACM 33 (5) (1990) 539–543. doi:10.1145/78607.78614

G. M. Amdahl, Validity of the single processor approach to achieving large scale computing capabilities, in: Proceedings of the April 18-20, 1967, Spring Joint Computer Conference, AFIPS ’67 (Spring), ACM, New York, NY, USA, 1967, pp. 483–485. doi:10.1145/1465482.1465560