COMPASS4: Space Charge Solvers

Sajid Ali

Scientific Computing Division
Fermi National Accelerator Laboratory

March 1, 2022

Outline

- Introduction
- Space charge solvers
- Implementing a finite difference space charge solver

Synergia2 overview

- `Synergia2`[†] is a particle accelerator simulator that uses the Particle-In-Cell technique[‡].
- Macro-particles represent a set of particles. A collection of macro-particles → bunch
- A simulator can contain two trains, with each train containing multiple bunches.
- A bunch is spread over multiple GPUs[§]. Particles are stored as SoA (Struct of Arrays) on each `MPI` rank as `Kokkos`[¶] views on the GPU. Host mirrors are used when dumping their state during checkpoint hooks.

[†] [Amundson 2006]

[‡] [Hockney 1988]

[§] We always map 1 `MPI` rank : 1 GPU in `Synergia2`

[¶] [Trott 2022]

Independent & Collective Operations

- Propagate is the primary routine that propagates bunch trains through lattices by applying steps.
- Steps contain independent & collective operators[†] . In this sub-section, we will concentrate on collective operators.
- The collective operator of interest here is a space-charge solver that is central to the PIC technique.
- This involves the following operations:
  - Deposit particles onto Cartesian grid.
  - Solve Poisson's equation on the grid.
  - Move the particles to their new locations as determined by the deduced force.

[†] Diagnostics are not considered here for simplicity

Communication avoiding concurrent solves

Current space charge solver: the Fourier Analysis Cyclic reduction scheme[†], executed concurrently over multiple sub-communicators, each of which solves the same equation with the same RHS.

Algorithm 1: FACR space charge solver

for *bunch in bunch-train* **do**
- all-ranks : deep-copy particles to host, `MPI_Allreduce` over all `MPI` ranks, deep-copy particles to device ;
- on each sub-communicator : FFT based solve ;
- all-ranks : deep-copy particles to host, `MPI_Allreduce` over `MPI` subcomm ranks, deep-copy particles to device ;

[†] [Swarztrauber 1977]

Need for a finite-difference solver

- The current solver only works for regular boundary conditions.
- We wish to more accurately model the space-charge effects in irregular accelerator elements.
- As a first pass attempt, we have opted to use the finite-difference method for simplicity.
- The size of the largest grid we plan to use is $256x256x1024$.
- The grid size is too small to effectively utilize all the GPUs for a typical simulation (beyond the strong scaling limit)[†].

[†] Heuristics by comparing results from [Li 2021], will fill in with some measured times here

Prototype in development

- Using PETSc's[†] `DMDA` for (scalable) distributed memory Cartesian grid data structures.
- Using dirichlet boundary conditions to set the potential at the boundary to be 0.
- We plan to apply boundary conditions either by having only a single diagonal entry ($= 1$) for all points outside the accelerator element (which still gives the matrix with the same dimensions as the corresponding free-space one) or by using `PetscSection` and filling only a subset of the matrix.
- Since the particles currently live on the GPU, we wish to solve the linear system there.

[†] [Balay 1997; Balay 2021a; Balay 2021b]

GPU solvers

- Direct solvers like `STRUMPACK`[†] and `SuperLU-dist`[‡] currently require one to pass the matrix and vectors from host memory and automatically offload LU factorization routines to the GPU themselves.
- Among iterative solvers that work directly on GPU matrices/vectors, we are exploring:
  - Multigrid preconditioners (`GAMG` & `hypre`, which use the CPU for solving on the coarse meshes).
  - Domain decomposition methods with sequential preconditioners.

[†] [Synk 2019]
[‡] [Sao 2019]

PETSc-SF for communication?

- PETScSF[†] (star-forest) is the communication component of PETSc and handles a multitude of communication patterns.
- Multiple back-ends (and the ability to choose one at runtime!) including host-only MPI, GPU-aware MPI and stream aware NVSHMEM[‡].
- We could use it for the allreduce & multi-broadcast communication pattern with device pointers when using >1 GPUs for the finite-difference space-charge solver.
- No-cost upgrade to stream-aware NVSHMEM back-end that allows us to explore the possibility of using concurrent streams for batches in the future!

[†] [Zhang 2021]

[‡] https://developer.nvidia.com/nvshmem

Initial results: solve times

Using the standard 5-point stencil and second-order finite difference discretization, for a grid size of $256x256x1024$; running on `NERSC-Perlmutter`, we have observed the following times (in sec):

Solver	1 GPU	2 GPU	4 GPU	8 GPU	16 GPU	32 GPU
ASM+ILU fact †	7.08e-1	3.78e-1	1.81e-1	9.38e-2	6.02e-2	N/A
CG+multigrid ‡	OOM	OOM	OOM	2.25e-3	1.34e-3	9.64e-4 §

The above results are with CUDA matrix/vector types (no CUDA code was used). Observed OOM errors with Kokkos matrix/vector types for the first solver and MPI crashes with the second solver. Moreover, kokkos-kernels (which provides portable ILU factorization routines) had build issues on FNAL-Wilson Cluster, so I was unable to test it extensively anywhere. Will report back once the issues are fixed.

† ASM+ILU refers to using the Additive Schwarz method [Widlund 1987] with Incomplete LU factorization on each block.

‡ Conjugate Gradient linear solver with GAMG multigrid preconditioning [Smith 2020]

§ GPU-aware MPI was not used, there are known issues when using GPU-aware MPI when large many MPI ranks on Perlmuter

Initial results: communication times

For a grid size of $256x256x1024$; measured communication times when using 64 GPUs in total, with 16 GPUs in each sub-communicator (running on `NERSC-Perlmutter`):

Method	Time (in sec)	notes
Baseline	2.265	redundant communication
PETScSF-cuda	0.915	with GPU-aware MPI
PETScSF-cuda	0.880	without GPU-aware MPI

Need to re-check whether we are missing some configuration with GPU-aware MPI as using it leads to no improvement. It could be that the current Cray-MPICH version has higher latency for GPU aware MPI and the time saved by not performing device-host/host-device transfers is lost in the higher latency. Note that the basic PETScSF type uses persistent point to point MPI operations.

Possible future upgrades

- Asynchronous[†] and Multi-precision[‡] GPU based solvers are currently being developed.
- We can harness these advances as and when they become available via their PETSc interface.

[†] [Nayak 2020; Zhang 2021]

[‡] [Abdelfattah 2020]

Acknowledgements

- FNAL Synergia2 team members.
- FNAL Wilson Cluster
- DoE-SCIDAC ComPASS-4
- NERSC Perlmutter

References I

[Abdelfattah 2020] Ahmad Abdelfattah, Hartwig Anzt, Erik G. Boman, Erin Carson, Terry Cojean, Jack Dongarra, Mark Gates, Thomas Grützmacher, Nicholas J. Higham, Sherry Li, Neil Lindquist, Yang Liu, Jennifer Loe, Piotr Luszczek, Pratik Nayak, Sri Pranesh, Siva Rajamanickam, Tobias Ribizel, Barry Smith, Kasia Swirydowicz, Stephen Thomas, Stanimire Tomov, Yaohung M. Tsai, Ichitaro Yamazaki, and Urike Meier Yang. *A Survey of Numerical Methods Utilizing Mixed Precision Arithmetic*. 2020. arXiv: 2007.06674 [cs.MS].

[Amundson 2006] J. Amundson, P. Spentzouris, J. Qiang, and R. Ryne. "Synergia: An accelerator modeling tool with 3-D space charge". *Journal of Computational Physics* 211.1 (2006), pp. 229–248. ISSN: 0021-9991.

[Balay 1997] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. "Efficient Management of Parallelism in Object Oriented Numerical Software Libraries". In: *Modern Software Tools in Scientific Computing*. Ed. by E. Arge, A. M. Bruaset, and H. P. Langtangen. Birkhäuser Press, 1997, pp. 163–202.

[Balay 2021a] Satish Balay, Shrirang Abhyankar, Mark F. Adams, Steven Benson, Jed Brown, Peter Brune, Kris Buschelman, Emil Constantinescu, Lisandro Dalcin, Alp Dener, Victor Eijkhout, William D. Gropp, Václav Hapla, Tobin Isaac, Pierre Jolivet, Dmitry Karpeev, Dinesh Kaushik, Matthew G. Knepley, Fande Kong, Scott Kruger, Dave A. May, Lois Curfman McInnes, Richard Tran Mills, Lawrence Mitchell, Todd Munson, Jose E. Roman, Karl Rupp, Patrick Sanan, Jason Sarich, Barry F. Smith, Stefano Zampini, Hong Zhang, Hong Zhang, and Junchao Zhang. *PETSc/TAO Users Manual*. Tech. rep. ANL-21/39 - Revision 3.16. Argonne National Laboratory, 2021.

References II

[Balay 2021b] Satish Balay, Shrirang Abhyankar, Mark F. Adams, Steven Benson, Jed Brown, Peter Brune, Kris Buschelman, Emil M. Constantinescu, Lisandro Dalcin, Alp Dener, Victor Eijkhout, William D. Gropp, Václav Hapla, Tobin Isaac, Pierre Jolivet, Dmitry Karpeev, Dinesh Kaushik, Matthew G. Knepley, Fande Kong, Scott Kruger, Dave A. May, Lois Curfman McInnes, Richard Tran Mills, Lawrence Mitchell, Todd Munson, Jose E. Roman, Karl Rupp, Patrick Sanan, Jason Sarich, Barry F. Smith, Stefano Zampini, Hong Zhang, Hong Zhang, and Junchao Zhang. *PETSc Web page*. https://petsc.org/. 2021.

[Hockney 1988] R. W. Hockney and J. W. Eastwood. *Computer Simulation Using Particles*. USA: Taylor & Francis, 1988. ISBN: 0852743920.

[Li 2021] Ruipeng Li and Ulrike Meier Yang. "Performance Evaluation of hypre Solvers". (Feb. 2021).

[Nayak 2020] Pratik Nayak, Terry Cojean, and Hartwig Anzt. *Evaluating Abstract Asynchronous Schwarz solvers on GPUs*. 2020. arXiv: 2003.05361 [cs.DC].

[Sao 2019] Piyush Sao, Xiaoye S. Li, and Richard Vuduc. "A communication-avoiding 3D algorithm for sparse LU factorization on heterogeneous systems". *Journal of Parallel and Distributed Computing* 131 (2019), pp. 218–234. ISSN: 0743-7315.

References III

[Smith 2020] Barry Smith, R T Mills, T Munson, S Wild, M Adams, S Balay, G Betrie, J Brown, A Dener, S Hudson, M Knepley, J Larson, O Marin, H Morgan, J-L Navarrow, K Rupp, P Sanan, H Zhang, H Zhang, and J Zhang. *Recent PETSc/TAO Enhancements for Exascale*. https://ecpannualmeeting.com/assets/overview/posters/petsc-ecp2020-poster.pdf. 2020.

[Swarztrauber 1977] Paul N. Swarztrauber. "The Methods of Cyclic Reduction, Fourier Analysis and the FACR Algorithm for the Discrete Solution of Poisson's Equation on a Rectangle". *SIAM Review* 19.3 (1977), pp. 490–501. ISSN: 00361445.

[Synk 2019] Ryan Synk and Pieter Ghysels. "A GPU-Accelerated Structurally-Symmetric Sparse Multifrontal Solver". (Aug. 2019).

[Trott 2022] Christian R. Trott, Damien Lebrun-Grandié, Daniel Arndt, Jan Ciesko, Vinh Dang, Nathan Ellingwood, Rahulkumar Gayatri, Evan Harvey, Daisy S. Hollman, Dan Ibanez, Nevin Liber, Jonathan Madsen, Jeff Miles, David Poliakoff, Amy Powell, Sivasankaran Rajamanickam, Mikael Simberg, Dan Sunderland, Bruno Turcksin, and Jeremiah Wilke. "Kokkos 3: Programming Model Extensions for the Exascale Era". *IEEE Transactions on Parallel and Distributed Systems* 33.4 (2022), pp. 805–817.

[Widlund 1987] Olof Widlund and M Dryja. *An additive variant of the Schwarz alternating method for the case of many subregions*. Technical Report 339, Ultracomputer Note 131. Department of Computer Science, Courant Institute, Dec. 1987.

References IV

[Zhang 2021] Junchao Zhang, Jed Brown, Satish Balay, Jacob Faibussowitsch, Matthew G. Knepley, Oana Marin, Richard Tran Mills, Todd S. Munson, Barry F. Smith, and Stefano Zampini. "The PetscSF Scalable Communication Layer". *CoRR* abs/2102.13018 (2021). arXiv: 2102.13018.