SELF-CONSISTENT PARALLEL MULTI BUNCH BEAM-BEAM SIMULATION USING A GRID-MULTIPOLE METHOD

F.W. Jones, TRIUMF, Vancouver, Canada
W. Herr and T. Pieloni, CERN, Geneva, Switzerland

Abstract

The simulation code COMBI has been developed to enable the study of coherent beam-beam effects in the full collision scenario of the LHC, with multiple bunches interacting at multiple crossing points over many turns. The parallel version of COMBI was first implemented using a soft-Gaussian collision model which entails minimal communication between worker processes. Recently we have extended the code to a fully self-consistent collision model using a Grid-Multipole method, which allows worker processes to exchange charge and field information in a compact form which minimizes communication overhead. In this paper we describe the Grid-Multipole technique and its adaptation to the parallel environment through pre- and post-processing of charge and grid data. Performance measurements in a Myrinet cluster environment will be given.

INTRODUCTION

The parallel beam-beam code COMBI[1] was developed to extend the reach of coherent beam-beam simulations to multiple bunches and multiple interaction regions, and thus to account for the different collision patterns experienced by bunches in the LHC and other colliders. These patterns are influenced by the bunch filling patterns (possibly different) in the two rings, and the location and number of interaction points (IPs) in operation. Moreover, around each crossing point will be grouped a series of parasitic collisions where bunches are in close proximity. The collision patterns define distinct equivalence classes for bunches, within which there will be unique coherent oscillation spectra and possible origins of instability.

In addition to the soft-Gaussian model of bunch interactions in COMBI, we have now implemented a fully self-consistent model based on Fast Multipole Methods (FMM) in a parallel "Grid-Multipole" algorithm. This hybrid of Particle-In-Cell and Fast-Multipole techniques has been previously applied to space charge[2] and single-IP beam-beam[3] simulations.

In the following we describe in the detail the parallel architecture and implementation of the Grid-Multipole calculation and evaluate its scalability and performance relative to the soft-Gaussian model.

STRUCTURE OF COMBI

The structure of the computational core is based on nested stepping over turns, bunches, and ring locations. Zero, one, or two bunches (one from each beam) may occupy a ring location ("slot") and to any location there may be assigned an *action code* that specifies a computation to be done. The book-keeping for these operations is managed by a *Supervisor Process* whereas the computations specified by the action codes are done in parallel by a number of *Worker Processes* (see Figure 1) . The Supervisor sends and receives MPI messages to and from all the Workers, whereas Workers communicate with each other only in pairs, as dictated by the head-on and parasitic collisions occurring in a given time-step.

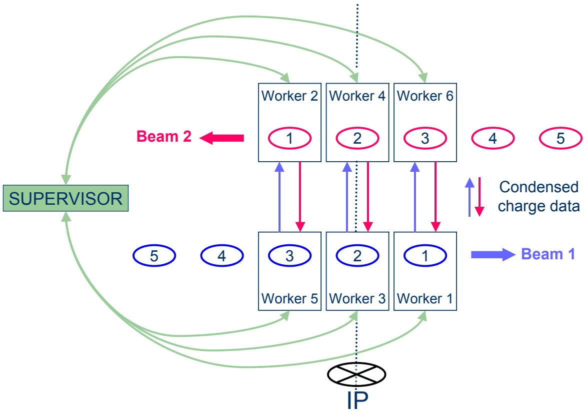

Figure 1: Parallel communication paths

In general, the worker processes handling a pair of interacting bunches must exchange either charge or field information, in order for each worker to determine the forces acting on each particle of the bunch and apply deflections (kicks) to its locally-stored coordinates. In the soft-Gaussian model, worker processes need only exchange the centroid and standard deviations of the charge distributions, resulting in very compact message data from which the forces can be calculated directly.

By contrast, a self-consistent model requires forces to be computed from the actual charge distributions of the bunches, involving $\sim 10^5$–10^6 macroparticles or up to $\sim$8 MB of coordinate data. Potentials or field components involve comparable volumes of data. The FMM solver is able to render a more compact representation of these quantities in terms of multipole coefficients, but these are bound to the internal quad-tree decomposition and other data structures in the solver. The solver currently used in COMBI is not intrinsically parallel and would require sweeping revisions to efficiently reproduce internal states across processes. For the present study we did not pursue parallelism at the FMM-internal level, although it may be considered in the future due to new parallel FMM implementations becoming available.

The Grid-Multipole technique, although originally adopted due to its ability to effectively model parasitic collisions with two separated charge distributions, also offers, through its pre- and post-processing, the means to efficiently exchange charge information between processes in an MPI setting.

Grid-Multipole Basics

As in Particle-In-Cell (PIC) simulations, the Grid-Multipole technique comprises three distinct stages:

1. Charge condensation (from discrete charges to grid points)
2. Condensed-charge field solution (FMM of grid-point charges)
3. Force distribution (interpolation of field components from grid points to discrete-charge locations)

We refer to charge "condensation" because the gridding of charges reduces the effective number of charges used in the FMM solution and hence the amount of data exchanged by processes. Unlike the conventional PIC method where all points on the grid must be included in a discrete FFT computation, only grid points containing charges are needed for the FMM solution, thus further reducing the data exchange volume. As is common practice, to reduce discretization error each source charge is distributed to the four nearest grid points by bilinear extrapolation and force components are distributed to the destination charge locations by the complementary interpolation. This "cloud-in-cell" charge condensation yields, for example, a factor of $\sim$94% reduction in the charge population for a macroparticle bunch of Gaussian form (truncated at 10σ) with a grid spacing of 0.28σ.

Grid-Multipole Processing and Data Management

When the FMM program option is enabled, the Supervisor sends to each Worker process information on the grid size and spacing as part of its "initialize" message. Subsequently, when a Worker receives a "collision" directive for a bunch that it manages, it performs the following actions (shown schematically in Figure 2):

1. Prepare a gridded charge interleaved data structure.
2. Send the data structure to the partner Worker process managing the opposing colliding bunch.
3. Receive the complementary gridded charge data structure from the partner process.
4. Assemble the FMM solver input and invoke the solver.
5. Unpack the FMM solver output (gridded field components), distribute forces to the local bunch's macroparticle locations, and apply the consequent angular deflections.
6. Send "action completed" message to the Supervisor.

The bunch coordinate arrays and charge grid array are local structures and gridding of charges is performed locally. All data to be sent to the partner bunch is assembled into a single MPI message in an interleaved fashion, allowing variable message size.

Generally the grid dimensions are chosen so as to enclose all charges of the local bunch and all received gridded-charge data of the partner bunch. However, this is not a strict requirement and any particles falling outside the grid, at any time during the simulation, are included in the FMM field computation as separate, single charges. These charges are included in the MPI message and are tagged as "halo particles" as indicated in Figure 2. They are also placed into the local data arrays for the FMM solution. When the gridding of local charges is complete, the grid data is interleaved into the MPI message as (I, J) indices and charge values.

A comparable message is received from the partner bunch process, and is unpacked to: (1) add the partner's halo charges to the FMM input arrays, and (2) add the partner's gridded charges to the local grid (the overlap of gridded charges will vary depending on the type of collision, crossing angle, and interaction-region optics). In the first instance, the FMM method computes potentials and fields only at the actual charge locations. For the beam-beam effect, unlike space charge effects, only the field of the opposing beam plays a role. Hence the local charges are diminished to vanishingly small values and serve only as place-holders in the FMM to recover the field values at the local charge locations.

Once local and partner grid data are merged, the grid charges are transferred to the FMM input arrays, transforming (I, J) grid indices into actual (X, Y) coordinates based on the grid spacing. After a charge is transferred, its grid cell is re-used to store the index of the charge in the FMM input arrays. This index is later used to retrieve the electric field components at each grid point from the E_x and E_y arrays returned by the FMM solver. Computation of the field components and beam-beam kicks at each bunch charge location is then a straightforward interpolation process, as shown in the last phase of Figure 2.

PERFORMANCE

Clearly the grid-multipole technique involves pre- and post-processing overheads compared to the soft-Gaussian model, although the extra processing time is generally much smaller[2] than the actual FMM solution time. The Gaussian model involves less preparation, a very small volume of message data, and direct field computation, so we performed timing trials for both Gaussian and FMM options as shown in Figure 3. The results for escalating numbers of collisions show that both methods scale well and that FMM is reasonably competitive. This is helped by the fact that the MPI traffic is dominated by one-to-one communications. The one-to-many and many-to-one transactions with the supervisor process involve only small amounts of data.

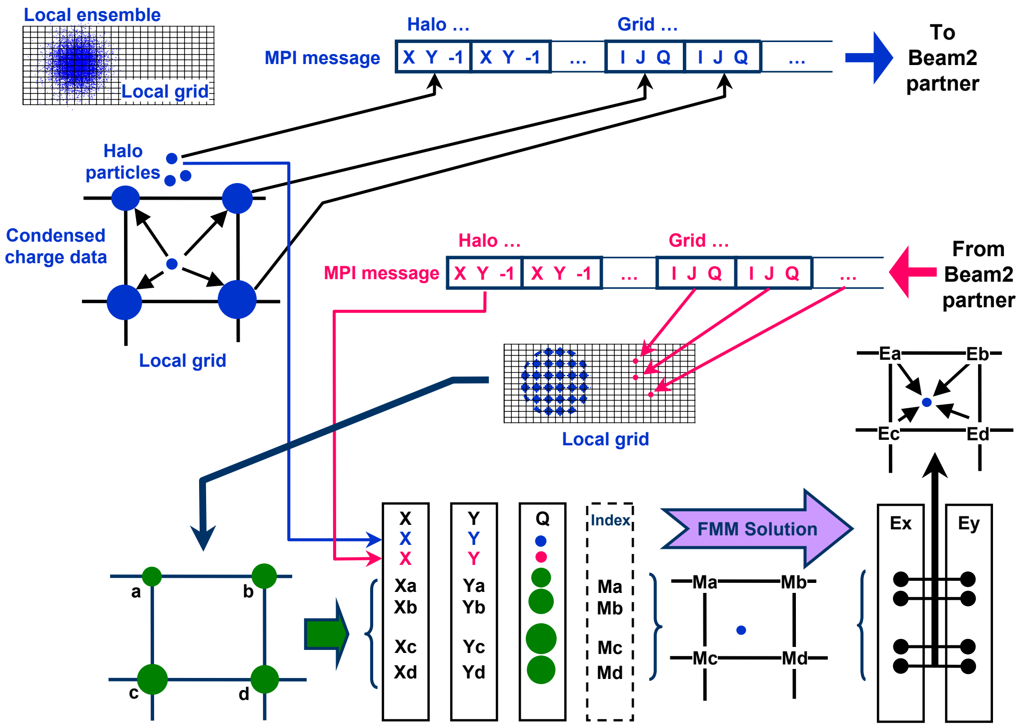

Figure 2: Parallel grid-multipole pre- and post-processing

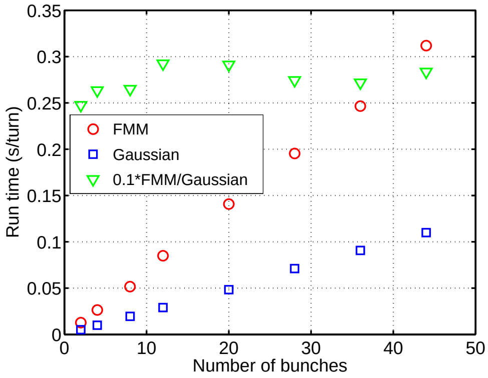

Figure 3: Parallel performance

EXAMPLE

In Figure 4 are shown the spectra produced by the two methods for a case using a simplified LHC lattice, with two trains of 3 bunches colliding at one IP. At the IP the beams experience a head-on collision and 1 parasitic collision on each side of the crossing point at a separation of 4σ.

The spectra agree qualitatively in the number and location of coherent modes. The restricted degrees of freedom in the Gaussian model result in an underestimation of the force[3] and consequently of the frequency spread, whereas the FMM method is unconstrained and gives a more accurate depiction of the spread and the π-mode frequencies.

CONCLUSION

The parallel multi-bunch multi-IP code has been extended to a self-consistent field computation. A grid-multipole technique is used which yields efficient MPI communication and good scaling properties. It gives a more accurate description than the soft-Gaussian model, at the expense of a factor ∼3 increase in computing time.

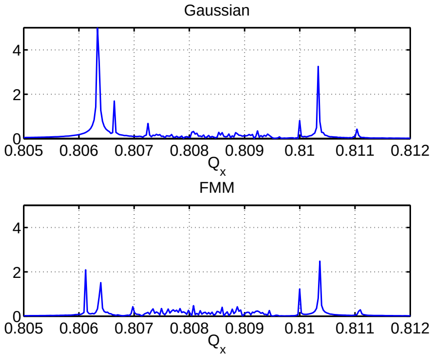

Figure 4: Spectra of coherent motion

The authors would like to thank École Polytechnique Fédérale de Lausanne, Drs. J. Menu and J.C. Leballeur, for access to excellent computing facilities at EPFL DIT-HPC.

REFERENCES

[1] F.W. Jones, W. Herr and T. Pieloni, Parallel Beam-Beam Simulation Incorporating Multiple Bunches and Multiple Interaction Regions, PAC 2007, Albuquerque NM.

[2] F.W. Jones, A Hybrid Fast-Multipole Technique for Space-Charge Tracking With Halos, Workshop on Space Charge Physics in High Intensity Hadron Rings, AIP Conf. Proceedings **448**, 1998.

[3] W. Herr, M.P. Zorzano and F. Jones, Hybrid Fast Multipole Methods Applied to Beam-Beam Collisions in the Strong-Strong Regime, Phys. Rev. ST Accel. Beams **4**, 054402 (2001).