PPM: A Scalable Software Framework for Parallel Particle-Mesh Simulations

Omar Awile, Milan Mitrović, and Ivo F. Sbalzarini
MOSAIC Group, Institute of Theoretical Computer Science, ETH Zurich, 8092 Zurich, Switzerland.
SIB Swiss Institute of Bioinformatics, 8092 Zurich, Switzerland.

The Parallel Particle Mesh (PPM) library is a parallelization layer for hybrid particle-mesh simulations based on MPI [1]. It provides an API infrastructure for transparently distributing the data and operations of hybrid particle-mesh simulations onto distributed-memory computers.

Hybrid particle-mesh methods provide a versatile simulation framework for both discrete and continuous models. When simulating discrete models, the computational particles directly correspond to the interacting entities. Examples include molecular dynamics, simulations of car traffic, and Monte Carlo methods. When simulating continuous models, such as PDEs, the particles correspond to Lagrangian tracer points of the continuous fields or to the collocation points of a mesh-less discretization scheme, such as smoothed particle hydrodynamics (SPH), reproducing kernel particle methods (RKPM), moving least squares, or particle strength exchange (PSE). Far-field interactions are efficiently computed on a regular Cartesian background mesh, leading to hybrid particle-mesh methods, such as vortex methods for incompressible fluid mechanics or PIC methods for plasma physics.

The PPM core implements parallel data and operation abstractions for particle-mesh methods [2]. Frequently used solvers are implemented in the PPM numerics package, but the user code (the “PPM client”) implements the specific numerics (see Fig. 1). The abstractions in PPM are on an intermediate level of granularity, making communication overhead explicit. No abstraction contains “hidden” internal communication and all communication is controlled by the client program.

The PPM core uses an object-oriented software architecture [3] and has recently been supplemented with multi-threading/multi-processing and GPU support for moment-conserving particle-mesh interpolation. Recent work has focused on developing a domain-specific language and a source-to-source compiler that translates workflow specifications in this language to compilable PPM client source code.

PPM has been used in diverse applications ranging from molecular dynamics to discrete element simulations of granular flows to continuum fluid mechanics, and PPM-based codes have frequently outperformed hand-parallelized benchmark codes [2].

![Diagram of PPM layer](image)

Figure 1: Scheme of the PPM layer sitting between MPI and the user’s client applications.

References