A Language and Development Environment for Parallel Particle Methods

Tobias Nett*, Sven Karol*, Jeronimo Castrillon* and Ivo F. Sbalzarini† ††

*Chair for Compiler Construction, Center for Advancing Electronics Dresden,
TU Dresden, Dresden, Germany
[tobias.nett | sven.karol | jeronimo.castrillon]@tu-dresden.de

† Chair of Scientific Computing for Systems Biology, Faculty
of Computer Science, TU Dresden, Dresden, Germany

†† MOSAIC Group, Center for Systems Biology Dresden,
Max Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany
ivos@mpi-cbg.de

ABSTRACT

We present the Parallel Particle-Mesh Environment (PPME), a domain-specific language (DSL) and development environment for numerical simulations using particles and hybrid particle-mesh methods. PPME is the successor of the Parallel Particle-Mesh Language (PPML) [1,2], a Fortran-based DSL that provides high-level abstractions for the development of distributed-memory particle-mesh simulations with the parallel particle-mesh library for high-performance computing [3]. The abstractions in PPML allow scientific programmers to write more concise and declarative code in comparison to hand-coded implementations. Essentially, it frees developers from the burden of writing boilerplate code that manages parallelism, synchronization, and data distribution. However, PPML has downsides which we address in PPME [4]: The lightweight embedding of PPML into Fortran, based on language macros, prevents advanced code analysis and complex compile-time computation. This makes debugging PPML programs hard and prohibits domain-specific static code optimization. PPME improves this by providing a complete development environment for particle-based simulations based on state-of-the-art language engineering and compiler construction techniques. Our contributions include a novel domain metamodel, which allows us to implement analysis and optimization algorithms that are well-suited for particle methods. The model is the basis of a formal type system with optional verification of physical dimensions. This enables advanced domain-specific correctness checks at compile time at the level of particle abstractions, complementing the low-level analysis of the compiler. We further show the optimization capabilities of PPME by adopting Herbie [5] for improving the accuracy of floating-point expressions and equations. Since PPME is integrated into the meta programming system (MPS) [6], it supports a convenient high-level mathematical notation for equations and differential operators. For demonstration purposes, we implemented several case studies that simulate discrete and continuous models using particle methods in PPME.

REFERENCES

[1] Omar Awile, Milan Mitrović, Sylvain Reboux, and Ivo F. Sbalzarini. 2013. "A domain-specific programming language for particle simulations on distributed-memory parallel computers." In *Proc. III Intl. Conf. Particle-based Methods (PARTICLES)*. Stuttgart, 436–447.

[2] Omar Awile. 2013. "A Domain-Specific Language and Scalable Middleware for Particle-Mesh Simulations on Heterogeneous Parallel Computers." PhD Thesis, Diss. ETH No. 20959. ETH Zürich.

[3] Ivo F. Sbalzarini, J. H. Walther, M. Bergdorf, S. E. Hieber, E. M. Kotsalis, and P. Koumoutsakos. 2006. "PPM - A highly efficient parallel particle-mesh library for the simulation of continuum systems." *J. Comput. Phys.* **215, 2** (2006), 566–588.

[4] Sven Karol, Pietro Incardona, Yaser Afshar, Ivo F. Sbalzarini, and Jeronimo Castrillon. 2015. "Towards a Next-Generation Parallel Particle-Mesh Language." In *Proc. of DSLDI'15.* 15–18.

[5] Pavel Panchekha, Alex Sachez-Stern, James R. Wilcox, and Zachary Tatlock. 2015. "Automatically Improving Accuracy for Floating Point Expressions." In *PLDI'15*, Vol. 50. ACM, 1–11.

[6] Markus Voelter. 2013. "Language and IDE Modularization and Composition with MPS." *In Gen. and Trans. Techn. in Soft. Eng. IV*, Vol. 7680. Springer, 383–430.