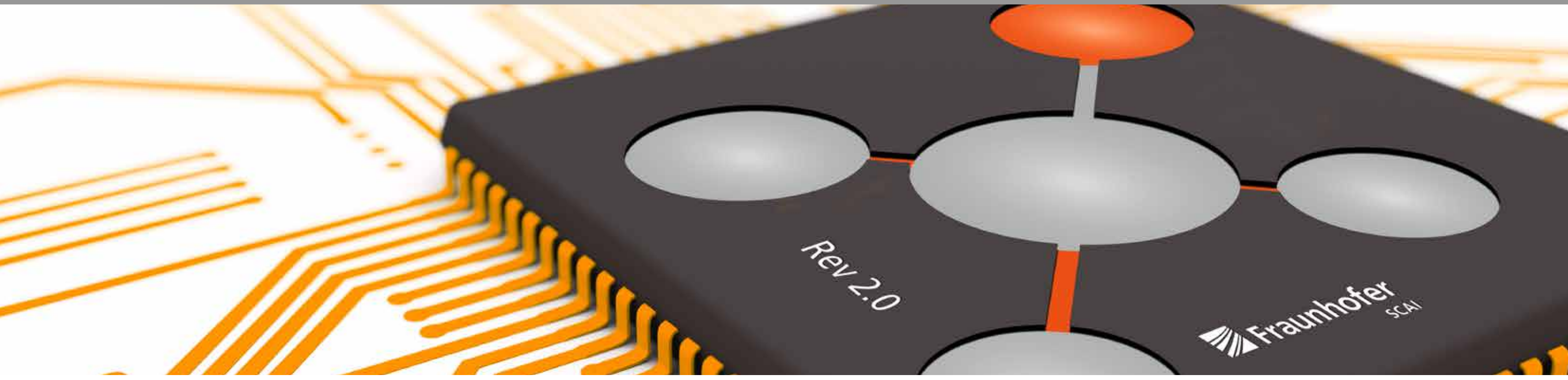
Fraunhofer SCAI

FRAUNHOFER INSTITUTE FOR ALGORITHMS AND SCIENTIFIC COMPUTING SCAI



LAMA

FAST HARDWARE INDEPENDENT SOFTWARE DEVELOPMENT

LAMA is a framework for developing hardware-independent, high performance code for heterogeneous computing systems. It facilitates the development of fast and scalable software that can be deployed on nearly every type of system, from embedded devices to highly parallel supercomputers, with a single code base. By using LAMA for their application, software developers benefit from higher productivity in the implementation phase and stay up to date with the latest hardware innovations, both leading to shorter time-to-market.

Overview

LAMA is a multi-layer framework (Fig. 1) offering multiple modules for fast heterogeneous software development. It targets multi-core CPUs, NVIDIA® GPUs and Intel® Xeon® Phi™'s – for single-node or multi-node usage, i.e. any kind of a distributed homogeneous or heterogeneous environment. LAMA's flexible plug-in architecture allows a seamless integration of tomorrow´s CPU´s and accelerator hardware architectures, thus reducing maintenance costs. The modular and extensible software design supports the developer on several levels, regardless of whether writing his own portable code with the Heterogeneous Computing Development Kit or using functionality from the higher level packages, the user always gains high productivity and maximum performance.

LAMA is written in C++ and the latest version 3.0 uses the new modern C++11 features that allow writing codes more securely and more efficiently. LAMA is licensed for free under LGPL (GNU Lesser General Public License v3), so derivative work must also be redistributed under LGPL, but applications using the LAMA library don't have to be.

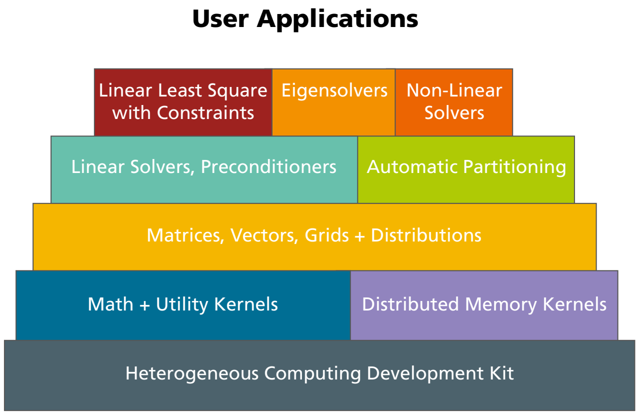
Fraunhofer Institute for Algorithms and Scientific Computing SCAI
Schloss Birlinghoven 1
53757 Sankt Augustin
Germany

Contact:
Dr. Thomas Soddemann
Phone +49 2241 14-4076
thomas.soddemann@
scai.fraunhofer.de

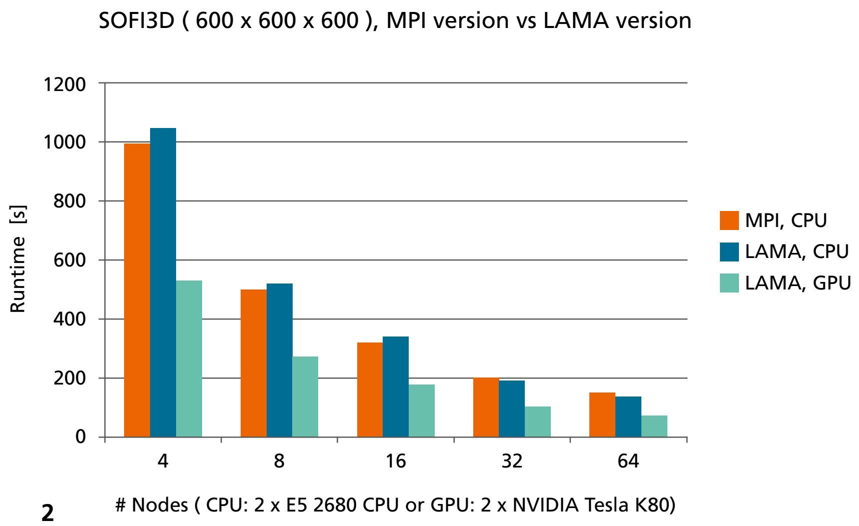
www.scai.fraunhofer.de/hpc

Website:
Find more information about LAMA on
www.libama.org

June 2018


User Applications
Linear Least Square with Constraints
Eigensolvers
Non-Linear Solvers
Linear Solvers, Preconditioners
Automatic Partitioning
Matrices, Vectors, Grids + Distributions
Math + Utility Kernels
Distributed Memory Kernels
Heterogeneous Computing Development Kit

1



2

1 *LAMA's multi-layer software stack*

2 *Performance Comparison between MPI and LAMA version of SOFI3D for a 600 x 600 x 600 grid.*

Heterogeneous Computing Development Kit

The Heterogeneous Computing Development Kit as LAMA's base module provides the management of heterogeneous memory and compute kernels. It facilitates three issues:

- A consistent data usage on heterogeneous devices is achieved by dedicated read and write accesses within the memory management.
- Decisions about the execution context in the application are separated from implementing the kernels by the kernel management.
- Asynchronous execution of these kernels, memory transfer and communication is handled by the tasking layer.

This combination leads to a clean software development, accomplishing a good maintainability on the user's side with minimal runtime overhead.

Kernel Layer

The Kernel Layer provides hardware independent routines for multiple purposes: basic functionality on arrays, dense and sparse BLAS operations as well as sparse conversion routines encapsulating MKL (BLAS), cuBLAS and cuSPARSE or own optimized kernels. Different sparse matrix formats are available for the application in various use cases and target architectures. This layer also manages different mappings of data to processor arrays and implements typical data parallel communication patterns that scale on cluster nodes. The kernels take direct advantage of any support for faster communication between device memory (e.g. CUDA-aware MPI).

Linear Algebra Layer

The Linear Algebra Layer facilitates the development of numerical algorithms for various application domains using vectors (sparse or dense), matrices (sparse or dense) or multi-dimensional grids distributed among the available processors. Operations can be written in textbook syntax where symbolic expression handling at runtime avoids temporaries as far as possible. Due to the underlying layers, all the operations here abstract from the used matrix/vector formats and distributions, and so hide implementation details of the target architecture, memory management and communication.

Application Layers

Based on the data structures of the Linear Algebra Layer, LAMA offers various iterative solvers using splitting methods (e.g.Jacobi, Richardson) or Krylov subspace methods (e.g. CG, GMRES, MINRES). They can be used either directly or preconditioned, with default or user-definable stopping criteria. Sparse matrix reordering and automatic partitioning can be applied to improve load balancing and to reduce the communication volume. Different implementations for eigensolvers, linear least square methods with constraints and non-linear solvers benefit from the LAMA approach and the available functionality. The integration of a custom-built solver is straightforward.

Case Study

SOFI3D is a seismic modelling code developed at the Geophysical Institute, KIT, Karlsruhe. The existing MPI version has been re-implemented with LAMA using explicit matrix-vector formalism. While the MPI version was difficult to maintain, the developers can now focus on geophysical problems and do not have to deal any more with implementation details and HPC issues. For a strong scaling benchmark, a 3D problem size with 600 grid points in each dimension has been selected. On the JURECA HPC system (Jülich Supercomputer Center) this benchmark shows nearly same performance for both versions on CPU nodes (2 x Intel Xeon E5 2680 v3 Haswell à 12 cores @ 2.5 GHz). In contrary to the MPI version, the LAMA version runs without modifications also on GPU nodes (2 x NVIDIA Tesla K80), see Fig. 2.