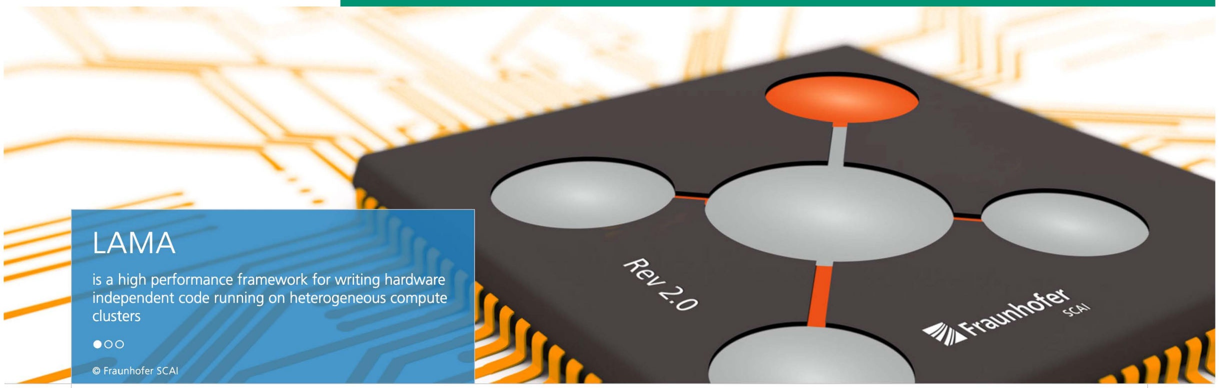


LAMA

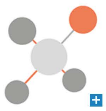
is a high performance framework for writing hardware independent code running on heterogeneous compute clusters

Core benefits

- hardware independent code for multiple platforms (Multicore, Nvidia® GPU, Intel® Xeon® Phi™)
- full cluster support (MPI, hybrid MPI parallelism)
- free Software distributed under GNU Lesser General Public License
- high productivity through highly portable code
- extendable design ensures support of latest hardware at all times

Overview

Framework



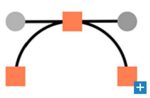
LAMA is a framework for developing hardware-independent, high performance code for heterogeneous computing systems. It facilitates the development of fast and scalable software that can be deployed on nearly every type of system, from embedded devices to highly parallel supercomputers, with a single 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.

Targets

The framework supports multiple target platforms (including GPUs and Xeon Phi) within a distributed heterogeneous environment. It offers optimized device code on the back-end side and high scalability through latency hiding and asynchronous execution across multiple nodes. LAMA's 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 prepared functionality from the Linear Algebra Package, the user always gains high productivity and maximum performance.

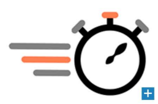
Design



LAMA's design enables its use on future hardware architectures with optimal performance ensured due to its inherent data structure layout that can be easily extended to support novel and even experimental hardware setups. LAMA includes unique communication features, which allow the data transfer between compute components within a node and between nodes to be completely hidden.

Performance

Productivity is combined with performance in execution – which is not mutually exclusive. LAMA's flexible software design introduces only a minimal overhead, conserving the full performance of the underlying BLAS implementations from the hardware vendors and from the highly optimized kernel back-ends. Performance comparison to concurring software libraries in the field of linear algebra show comparable results for single node implementations. On distributed systems the asynchronous execution model guarantees efficient overlapping of calculation, memory transfer and communication reaching linear scaling on GPUs.



Linear Algebra Package

The Linear Algebra Package facilitates the development of (sparse) numerical algorithms for various application domains. Code can be written in text-book-syntax as

y = A * x

(where x and y are vectors and A is a matrix). Due to the underlying layers, the problem formulation is handled independently of the implementation details regardless of the target architecture and distribution strategy as memory management and communication is processed internally. Furthermore, with load balancing between different components and asynchronous execution, full system performance can be obtained.

In addition, LAMA offers various iterative solvers like Jacobi or CG methods, that can be used directly or preconditioned, with a combination of several user-definable stopping criteria. Furthermore, the integration of a custom-built solver is straightforward.

Areas of Application

The target applications for LAMA are based mainly in High Performance Computing or Embedded Computing but can be wherever hardware independant applications are needed. The field of applications is huge, e.g., simulation as reservoir simulations, seismic imaging, performance engineering, or computational fluid dynamics but also image and video processing and many more.

Free Software

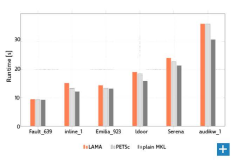
LAMA is licensed for free under LGPL (GNU Lesser General Public License v3), so derivative work must also be redistributed under LGPL, but applications using the LAMA library don't have to be.

How to get it

Your find a tar-ball of LAMA's release 3.0 here .

Performance

SINGLE-NODE CPU performance comparison



Comparison between LAMA, PETSc and a plain MKL BLAS implementation of an CG solver running 1000 iterations

System

- 6 MPI-processes on Intel® Xeon® E5-1650v2 (64GB DDR3 RAM)
- 6 random matrices from the University of Florida Collection
- CSR format
- Both libraries make use of Intel®'s high performance MKL BLAS implementation

Results

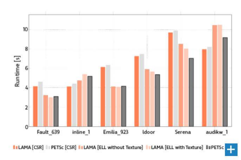
- Runtime is proportional to the number of non-zeros
- only the irregular structure of inline_1 and audikw_1 show remarkably higher runtime
- demonstrating, that LAMA's as well as PETSc's design overhead is negligible

In Summary

- LAMA and PETSc perform similar on CPUs

SINGLE-NODE GPU performance comparison

Comparison between LAMA and PETSc implementations of an CG solver running 1000 iterations



System

- Nvidia® K40 (12GB GDDR 5)
- CSR and ELL format

CSR format results

- the run time proportional to the number of non-zeros
- irregular structure of inline_1 and audikw_1 leads to higher runtime

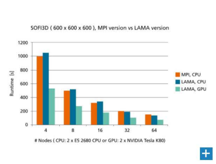
ELL format results

- show shorter run times in general
- except inline_1 and audikw_1 exhibiting nearly twice the number of entries per row compared to the other matrices

In Summary

- for the CSR format
  - LAMA and PETSc perform similar with a tiny overall benefit in favor of LAMA
  - both libraries rely on cuSPARSE SpMV implementation (dominating with 80% of the overall runtime)
  - LAMA calls cuBLAS routines for the axpy and dot operations while PETSc exploits implementations using the Thrust library
- for the ELL format
  - the runtime results are more sensitive to the actual sparse matrix structure in comparison with CSR
  - LAMA uses a custom kernel
  - exploiting texture cache
  - increases the performance slightly in most cases

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.

LAMA White Paper

LAMA White Paper

You can download our latest white paper on LAMA, its design, implementation, and performance right here .

LAMA in the Press - Publications

Brandes, Th. and Schricker, E. and Soddemann, Th.: The LAMA Approach for Writing Portable Applications on Heterogeneous Architectures - Projects and Products of Fraunhofer SCAI, 2017, DOI :http://www.springer.com/de/book/9783319624570

Süß, Tim; Döring, Nils; Gad, Ramy; Nagel, Lars; Brinkmann, André; Feld, Dustin; Schricker, Eric; Soddemann, Thomas; Impact of the Scheduling Strategy in Heterogeneous Systems That Provide Co-Scheduling, in Proceedings of the 1st COSH Workshop on Co-Scheduling of HPC Applications, 2016, DOI: 10.14459/2016md1286954

Förster, M., Kraus, J.: Scalable parallel AMG on cc-NUMA machines with OpenMP. In: Computer Science - Research and Development, 2011, Volume 26, Issue 3-4, pp 221-228, DOI: 10.1007/s00450-011-0159-z

Kraus, J., Förster, M.: Efficient AMG on Heterogeneous Systems. In: Facing the Multicore Challenge II, Lecture Notes in Computer Science, 2012, Volume 7174, pp 133-146, DOI: 10.1007/978-3-642-30397-5_12

Kraus, J., Förster, M., Brandes, T., Soddemann, T.: Using LAMA for efficient AMG on hybrid clusters, Computer Science - Research and Development, 2013, Volumn 28, Issue 2-3, pp 211-220, DOI: 10.1007/s00450-012-0223-3

About

LAMA in public-funded projects

WAVE

- unded by the BMBF, this project is about accelerating acoustic wave propagation and time reversal algorithms as used in seismic imaging
- partners: KIT, TEEC, Fraunhofer SCAI

FAST

- funded by the BMBF, this project aims at enhancing the execution of algorithms on modern large computing systems by clever and fault tolerant scheduling. LAMA is a use case
- partners: University of Mainz, Technical University Munich, University of Cologne, Megware, Partec

MACH

- ITEA project funded by the national funding agencies (BMBF for Germany)
- 16 partners from Belgium, France, Germany and The Netherlands
- It is about bridging the gap between traditional high performance computing and embedded computing

Past Projects

ENHANCE, GASPI, MWARE

How to get in touch with us

For any questions on LAMA please contact us via:

lama[at]scai.fraunhofer.de