List of Project Topics (Proposals)

Last update: Oct 22, 2025

LDE Projects in the Context of Our Two Open-source Systems

- **DAPHNE EU-project**
  https://github.com/daphne-eu/daphne
  - Focus on integrated data analysis pipelines
  - Project implementation mainly in C++

- **Apache SystemDS**
  https://github.com/apache/systemds
  - Focus on the end-to-end data science lifecycle
  - Project implementation mainly in Java and DML

Topics in DAPHNE

Implementation mainly in C++ and DaphneDSL, some also in Python

https://github.com/daphne-eu/daphne/issues?q=state%3Aopen%20label%3A%22LDE%20winter%202025%2F26%22

#939 Efficient Join Ordering Using the UES Method

- **Motivation**
  - Finding an efficient join order is one of the most important and most challenging tasks in relational query optimization and various methods have been proposed in the literature
  - The UES method, which is based on upper bounds and sampling, is particularly simple yet effective
- **Task** (in C++)
  - Implement the UES method for join enumeration as a compiler pass in DAPHNE
  - Evaluate the pass on well-known community benchmarks (e.g., the Join Order Benchmark)
- **More Information & Hints**
  - https://github.com/daphne-eu/daphne/issues/939
  - Contact: Patrick Damme

#511 Efficient Parallel Hash-Join Operator for Speeding up the SSB

- **Motivation**
  - The Star Schema Benchmark (SSB) is a well-known benchmark for analytical query processing.
  - The runtime of most SSB queries is dominated by PK-FK joins and semi-joins.
  - An efficient join implementation is crucial for achieving good results in this benchmark.
- **Task** (in C++)
  - Implement an efficient parallel hash-join (separate build and probe) operator on columnar data.
  - Devise an efficient hash table implementation and support it for intermediate results in DaphneIR.
  - Parallelism should be achieved through multiple threads (MIMD), and optionally also through SIMD operations.
  - In a first step, multi-threading could be applied inside the operator; based upon that, an integration with DAPHNE's vectorized engine could be tackled.
- **More Information & Hints**
  - https://github.com/daphne-eu/daphne/issues/511
  - Contact: Patrick Damme

#987 Advanced Sparsity Estimation

- **Motivation**
  - Sparse matrices that contain mostly zero values are commonplace in many application domains and frequently arise as intermediate results during the processing of ML algorithms
  - By using sparse data representations and sparsity-exploiting kernels, the memory footprint and runtime of a data analysis workload can be reduced asymptotically
  - While sparsity estimation has been an active field of research for many years, DAPHNE currently only supports the simplest form, naive meta data estimators
- **Task** (in C++)
  - Augment DAPHNE's existing naive sparsity estimation by more sophisticated methods from the literature
  - Based on the resulting, more accurate sparsity estimates, make DAPHNE automatically select a dense or sparse (CSR) representation to optimize the processing w.r.t. memory footprint and/or runtime
  - Show the impact of the improved sparsity estimation on the performance of typical ML and/or graph processing workloads
- **More Information & Hints**
  - https://github.com/daphne-eu/daphne/issues/987
  - Contact: Patrick Damme

#521 Efficient Matrix Multiplication for Generic Value Types

- **Motivation**
  - Matrix multiplication is a central operation in many machine learning and data analysis algorithms.
  - Various libraries (e.g., BLAS) provide highly optimized implementations, but are typically limited to certain data and value types, especially dense matrices of single/double-precision floating point values.
  - DAPHNE strives to be extensible w.r.t. to data and value types, thus it needs efficient matrix multiplications for other types, too.
- **Task** (in C++)
  - Implement efficient matrix multiplication for various combinations of dense/sparse inputs/output, different values types (e.g., integers, bool), and input shapes (e.g., matrix-matrix and matrix-vector).
  - On the one hand, write hand-tuned kernels for a handful of cases
  - On the other hand, devise a generic implementation that comes as close as possible to these specialized ones.
  - Showcase the benefit of your kernels on ML algorithms dominated by matrix multiplications.
- **More Information & Hints**
  - https://github.com/daphne-eu/daphne/issues/521
  - Contact: Patrick Damme

#986 DAPHNE and the Python Data Science Ecosystem: Efficient String Data Transfer

- **Motivation**
  - As Python is the language of choice for most data scientists nowadays, DAPHNE supports efficient bi-directional data transfer with popular Python libraries like numpy, pandas, SciPy, TensorFlow, and PyTorch
  - So far, only the transfer of numeric data is supported
  - However, real-world data sets often contain string-valued attributes that need to be transformed to numbers before applying ML algorithms and this transformation could happen either in Python or in DAPHNE
- **Task** (in C++ and Python)
  - Extend DAPHNE's data transfer to/from Python (e.g., numpy, pandas) by efficient support for string data
  - Investigate if performing typical feature transformations like recoding and one-hot-encoding is more efficiently done in Python or in DAPHNE and optimize the transformations in DAPHNE
- **More Information & Hints**
  - https://github.com/daphne-eu/daphne/issues/986
  - Contact: Patrick Damme

#985 Efficient File I/O Plug-ins for Widely-used File Formats

- **Motivation**
  - The input to integrated data analysis pipelines, that combine query processing, machine learning, and high-performance computing, could be provided in various general-purpose and domain-specific file formats
  - To embrace a large variety of such formats, DAPHNE is extensible w.r.t. to file readers/writers, i.e., expert users can add their own file I/O plug-ins without touching the source code of DAPHNE
  - However, so far there is only a limited number of plug-ins available
- **Task** (in C++)
  - Implement file I/O plug-ins for a range of widely used file formats for different data modalities, such as tabular data, (sparse) matrices/graphs, audio, images, and time series
  - These plug-ins may be based on existing open-source libraries (with compatible license)
  - Apply format-specific and format-agnostic tricks to make the file I/O efficient, e.g., by exploiting parallelism or pushing down certain operations into the readers
- **More Information & Hints**
  - https://github.com/daphne-eu/daphne/issues/985
  - Contact: Patrick Damme

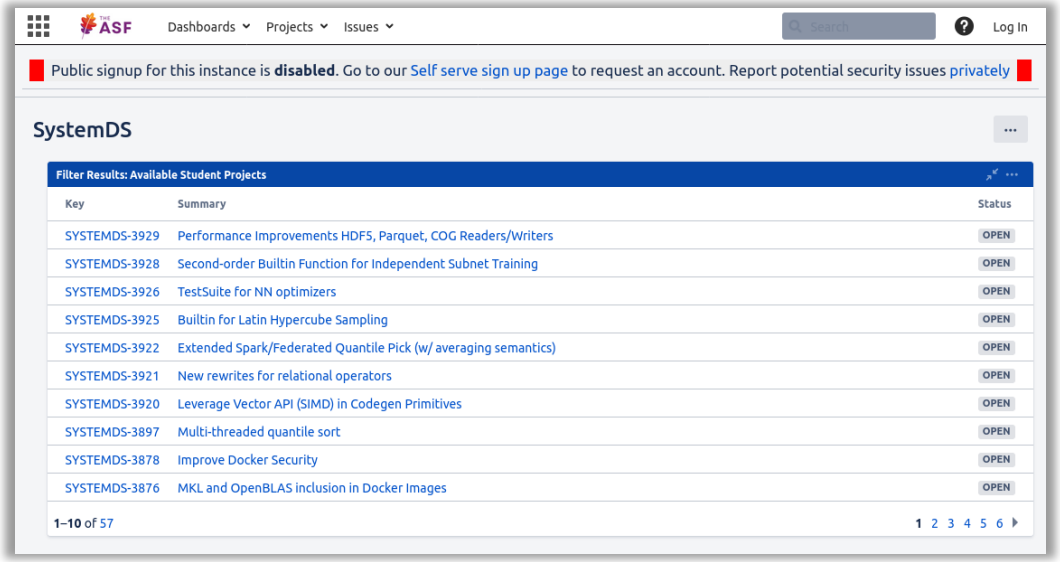
Topics in Apache SystemDS

Implementation mainly in Java and DML

https://issues.apache.org/jira/secure/Dashboard.jspa?selectPageId=12335852#Filter-Results/12365413

Topics in Apache SystemDS

- **See the Full List of Available Student Projects:**
  - https://issues.apache.org/jira/secure/Dashboard.jspa?selectPageId=12335852#Filter-Results/12365413



- **Topics Newly Added in WiSe 2025/26**
  - #3929 Performance Improvements HDF5, Parquet, COG Readers/Writers
  - #3928 Second-order Builtin Function for Independent Subnet Training
  - #3926 TestSuite for NN optimizers
  - #3925 Builtin for Latin Hypercube Sampling
  - #3922 Extended Spark/Federated Quantile Pick (w/ averaging semantics)
  - #3921 New rewrites for relational operators
  - #3920 Leverage Vector API (SIMD) in Codegen Primitives
- **More Information & Hints**
  - Contact: Matthias Boehm

Topics in TerseTS

Implementation mainly in Zig
https://github.com/cmcuza/TerseTS

Gorilla vs Chimp: Implementation and Evaluation of Lossless Time Series Compression in TerseTS

- **Task**
  - Implement the lossless time series compression methods Gorilla [1] and Chimp [2] in TerseTS
  - Additionally, systematically evaluate these compressors under varying data characteristics (e.g., frequency, volatility, dimensionality)
  - Provide insights into which compressor is best suited for different types of time series and inform the design of future compression systems for storage, transmission, and analytics
- **More Information & Hints**
  - Contact: Carlos E. Muniz Cuza
  - [1] Pelkonen et al.: Gorilla: A Fast, Scalable, In-Memory Time Series Database (PVLDB 2015)
  - [2] Liakos et al.: Chimp: Efficient Lossless Floating-Point Compression for Time Series Databases (PVLDB 2022)

Alternative: Propose Your Own Topic Idea

- **We are open to additional topic proposals**
  - In the context of data engineering, data management, and machine learning systems
  - If you are passionate about your idea
  - More topics in SystemsDS and DAPHNE or other open-source systems possible, but contributions might be more difficult to get accepted
  - **If you would like to propose your own topic**, **approach me by email by Oct 31, 23:59**; in any case, **also fill in the poll regarding the topic selection with your preferred topics from the list above**