Project Large-scale Data Engineering (LDE) Kick-off Meeting

Dr.-Ing. Patrick Damme
Technische Universität Berlin
Berlin Institute for the Foundations of Learning and Data
Big Data Engineering (DAMS Lab)

Last update: Apr 18, 2023

Announcements/Org

- **Hybrid Setting with Optional Attendance**
  - In-person in TEL 811 (~20 seats)
  - Virtual via zoom
    https://tu-berlin.zoom.us/j/67376691490?pwd=NmlvWTM5VUVWRjU0UGI2bXhBVkxzQT09

Agenda

- FG Big Data Engineering (DAMS Lab)
- Course Organization, Outline, and Deliverables
- Apache SystemDS
- DAPHNE
- List of Project Topics (Proposals)

FG Big Data Engineering (DAMS Lab)

https://www.tu.berlin/dams

FG Big Data Engineering (DAMS Lab) – Team

- **Head of the Group**
  Matthias Boehm

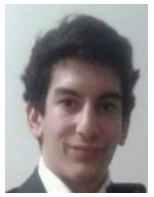
- **Postdoc** (01/2021)
  Patrick Damme

- **PhD Student** (04/2019)
  Arnab Phani

- **PhD Student** (01/2020)
  Sebastian Baunsgaard

- **PhD Student** (06/2023)
  Christina Dionysio

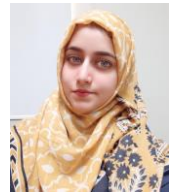
- **PhD Student** (06/2023)
  Philipp Ortner



- **PhD Student** (03/2023)
  David Justen
  SAP

- **PhD Student** (02/2023)
  Carlos E. Muniz Cuza
  [visitor Aalborg University]

- **PhD Student** (09/2019)
  Shafaq Siddiqi
  TU Graz



- **PhD Student** (04/2021)
  Saeed Fathollahzadeh
  TU Graz

- **3x Student Assistants**
  N.N. (1x ~06/2023)
- **Bachelor & Master Students**

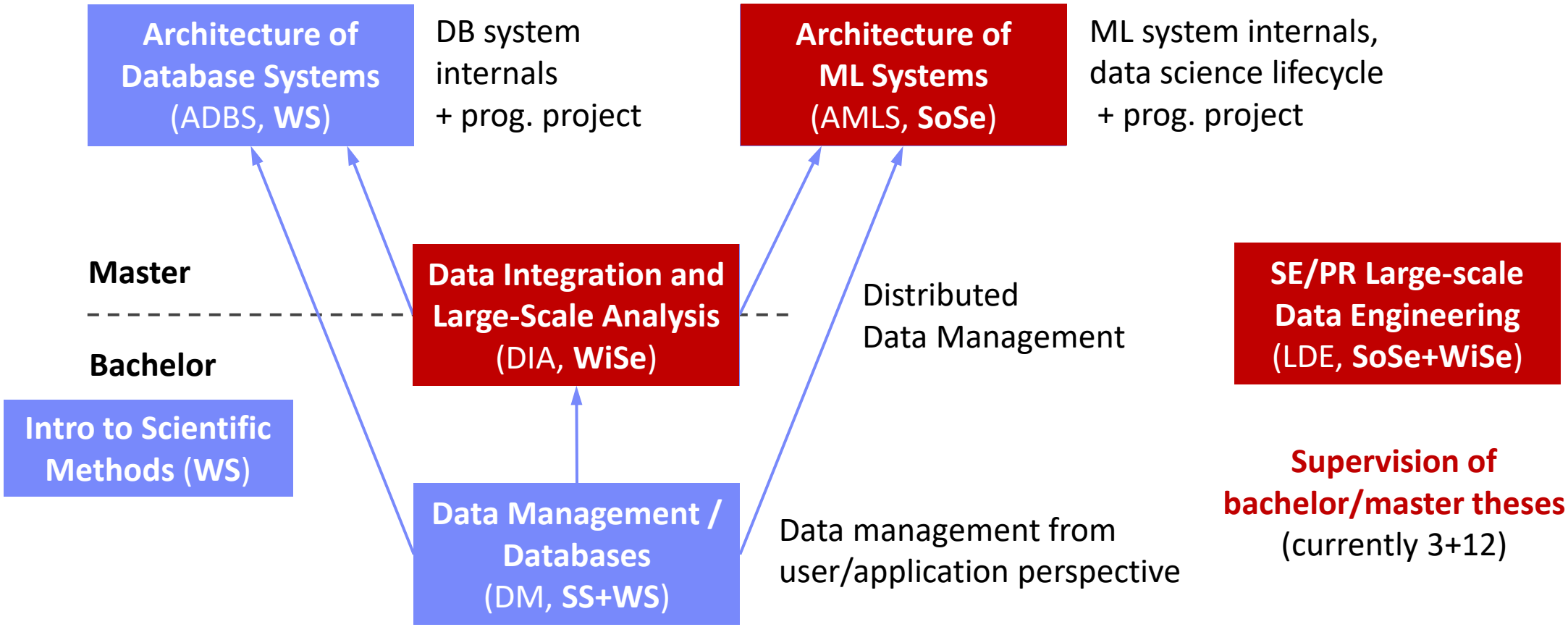
About Me

- **Since 10/2022: Postdoc at TU Berlin**, Germany
  - FG Big Data Engineering (DAMS Lab) headed by Prof. Matthias Böhm
  - Continuing work on integrated data analysis pipelines
  - Research interests in the fields of database and ML systems (especially compiler & runtime techniques, extensibility)

- **2021-2022: Postdoc at TU Graz & Know-Center GmbH**, Austria
  - Data Management group headed by Prof. Matthias Böhm
  - Started work on integrated data analysis pipelines

- **2015-2020: PhD student at TU Dresden**, Germany
  - Dresden Database Research Group headed by Prof. Wolfgang Lehner
  - PhD thesis on making complex analytical database queries more efficient through lightweight compression of intermediate results

FG Big Data Engineering (DAMS Lab) – Teaching



Course Organization, Outline, and Deliverables

Large-scale Data Engineering: Module Overview

20+5 places in total

bachelor + master

#41086: LDE Seminar + Project (12 ECTS)

13–2 students

12 students

#41095: Seminar LDE (3)

#41094: Project LDE (9 ECTS)

12+2 students

bachelor-only

bachelor-only

Mon, 14:00-16:00
TEL 811 & zoom

Seminar LDE

- Reading & writing scientific papers
- Giving presentations on papers
- Summary paper
- Presentation
- Lecturer & seminar mentor

Project LDE

- Building & evaluating prototypes
- Giving presentations on prototypes
- Prototype design/impl/tests/doc
- Presentation
- Project mentors




Mon, 16:00-18:00
TEL 811 & zoom

→ In the context of systems for data engineering, data management, machine learning

→ In combination: Ideal preparation for a bachelor/master thesis with our group

Course Organization

- **General Contact Person**
  - Dr.-Ing. Patrick Damme (patrick.damme@tu-berlin.de)

- **Course Website**
  - https://pdamme.github.io/teaching/2023_summer/lde/lde_summer2023.html
  - One site for seminar and project
  - All material, news, **deadlines**, …

- **Language**
  - Lectures and slides: **English**
  - Communication: **English**/**German**
  - Submitted paper and presentation: **English**
  - **Informal language** (first name is fine), immediate feedback is welcome

Semester Schedule & Deadlines

- **Kick-off Meeting Apr 17** (optional)
- **Self-organized Project Work**
  - Office hours for any questions (optional)
- **Final Presentations** (mandatory)
  - Individual appointments per student/team

- **Programmierpraktikum: Poster** (optional)
  - Centrally organized poster session in June
  - Each module will send 1-3 student teams
  - Great chance to showcase your work to your fellow students and future participants

- **List of Project Topics**
  - Presented today, take your time to select afterwards
- **Topic Selection**
  - **Deadline: May 1, 23:59 CEST** (in 2 weeks)
  - **First-come-first-serve** or **team building**
  - List of preferences by email to Patrick Damme
  - Feel free to approach us as a team
- **Submission of Impl/Tests/Doc**
  - **Deadline: Jul 24** (soft deadline)
  - As a pull request on GitHub (exceptionally by email)
- **Submission of Presentation Slides**
  - **Deadline: The day before you present, 23:59 CEST**
  - Presentation slides (PDF) by email to Patrick Damme

Project Deliverables

- **Individual/Team Project Work**
  - Teams of up to 3 students
- **Design/Implementation/Tests/Documentation**
  - Get familiar with the given task/problem
  - Develop an initial design for discussion
  - Discuss the design during the office ours **(strongly recommended)**
  - Implement your design, plus tests and docs
  - Conduct experiments and analyze/visualize results
- **Presentation**
  - Summarize the problem and your solution (design, implementation, experimental results)
  - **10 min talk + 5 min discussion**
- **Grading**
  - **#41086 (seminar + project)**
    - Graded portfolio exam
    - 25 pts: summary paper
    - 15 pts: presentation
    - 50 pts: design/impl/tests/doc
    - 10 pts: presentation
  - **#41094 (project-only)**
    - **Ungraded** portfolio exam
    - 85 pts: implementation/tests/documentation
    - 15 pts: presentation

LDE Project Characteristics

- **Unique Characteristics**
  - Each student/team gets a different topic
- **Advantages**
  - Topics are real open issues in existing systems
  - Meaningful contributions to open-source systems
  - Your work will be used by others (impact)
  - You earn 9 ECTS (~270 h of work)
    Programmierpraktikum: more than the usual 6 ECTS
- **Practice Open-source Processes**
  - Breakdown into subtasks
  - Code/tests/docs
  - Pull requests
  - Code review
  - Incorporate feedback to improve code
- **Remarks on Topic Descriptions**
  - Lots of open topics to work on in the two systems we develop in our group
  - Initial topic descriptions of varying level of detail
  - If there is interest in a specific topic, we will provide more detailed descriptions with some pointers
  - Open to alternative topic proposals

LDE Projects in the Context of Two Open-source Systems

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

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

Apache SystemDS: A Declarative ML System for the End-to-End Data Science Lifecycle

https://github.com/apache/systemds

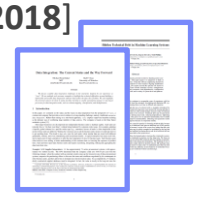
Landscape of ML Systems

- **Existing ML Systems**
  - **#1** Numerical computing frameworks
  - **#2** ML Algorithm libraries (local, large-scale)
  - **#3** Linear algebra ML systems (large-scale)
  - **#4** Deep neural network (DNN) frameworks
  - **#5** Model management, and deployment

- **Exploratory Data-Science Lifecycle**
  - **Open-ended problems** w/ underspecified objectives
  - Hypotheses, data integration, run analytics
  - Unknown value → lack of system infrastructure
    → **Redundancy of manual efforts and computation**

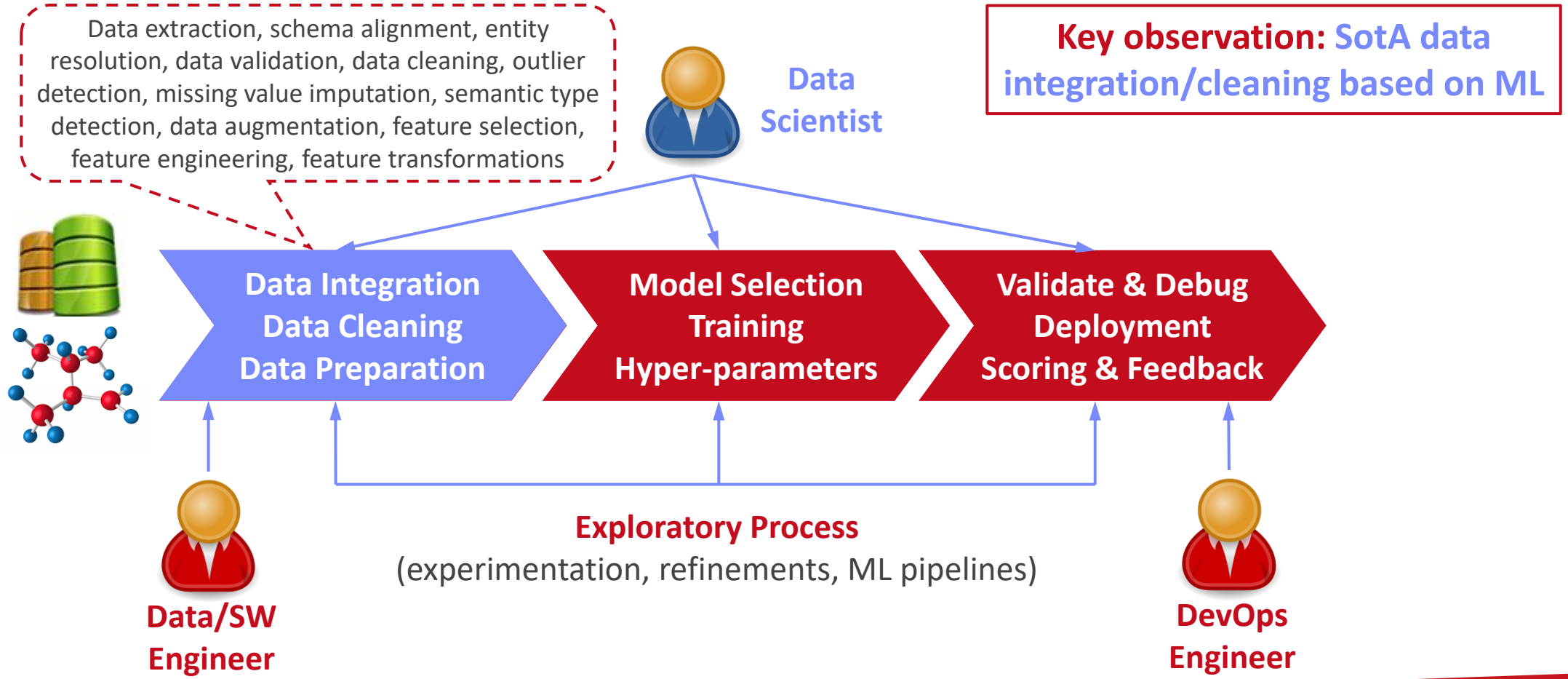
"Take these datasets and show value or competitive advantage"

[NIPS 2015]
[DEBull 2018]



- **Data Preparation Problem**
  - **80% Argument:** 80-90% time for finding, integrating, cleaning data
  - Diversity of tools → boundary crossing, lack of optimization

The Data Science Lifecycle (aka KDD Process, aka CRISP-DM)



Apache SystemDS [https://github.com/apache/systemds]

APIs: Command line, JMLC, Python Spark MLContext, Spark ML, (Scalable Algorithms + Primitives)

DML Scripts

Language

Compiler

Runtime

Write Once, Run Anywhere

[SIGMOD'15,'17,'19,'21abc,'23abc]
[PVLDB'14,'16ab,'18,'22]
[ICDE'11,'12,'15]
[CIDR'17,'20]
[VLDBJ'18]
[CIKM'22]
[DEBull'14]
[PPoPP'15]

Apache SystemML™

07/2020 Renamed to **Apache SystemDS**
05/2017 Apache Top-Level Project
11/2015 Apache Incubator Project
08/2015 Open Source Release

In-Memory Single Node
(scale-up)

In-Progress:

GPU

since 2014/16

Java

since 2012

Hadoop or Spark Cluster
(scale-out)

since 2010/11

since 2015

Federated
(LA progs, PS)

since 2019

Language Abstractions and APIs

Data Independence + Impl-Agnostic Ops
➔ **"Separation of Concerns"**

- **Example:** Stepwise Linear Regression

User Script

```
X = read('features.csv')
Y = read('labels.csv')
[B,S] = steplm(X, Y,
  icpt=0, reg=0.001)
write(B, 'model.txt')
```

Built-in Functions

```
m_steplm = function(...) {
  while( continue ) {
    parfor( i in 1:n ) {
      if( !fixed[1,i] ) {
        Xi = cbind(Xg, X[,i])
        B[,i] = lm(Xi, y, ...)
    } }
    # add best to Xg
    # (AIC)
} }
```

Feature Selection

```
m_lm = function(...) {
  if( ncol(X) > 1024 )
    B = lmCG(X, y, ...)
  else
    B = lmDS(X, y, ...)
}
```

ML Algorithms

```
m_lmCG = function(...) {
  while( i<maxi&nr2>tgt ) {
    q = (t(X) %*% (X %*% p))
      + lambda * p
    beta = ... }
}
```

```
m_lmDS = function(...) {
  l = matrix(reg,ncol(X),1)
  A = t(X) %*% X + diag(l)
  b = t(X) %*% y
  beta = solve(A, b) ...}
```

Linear Algebra Programs

Facilitates optimization across data science lifecycle tasks

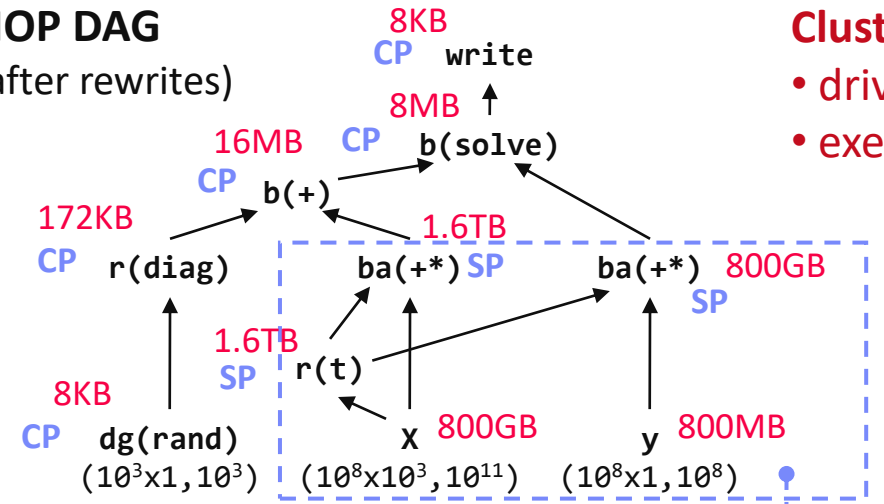
Basic HOP and LOP DAG Compilation

LinregDS (Direct Solve)

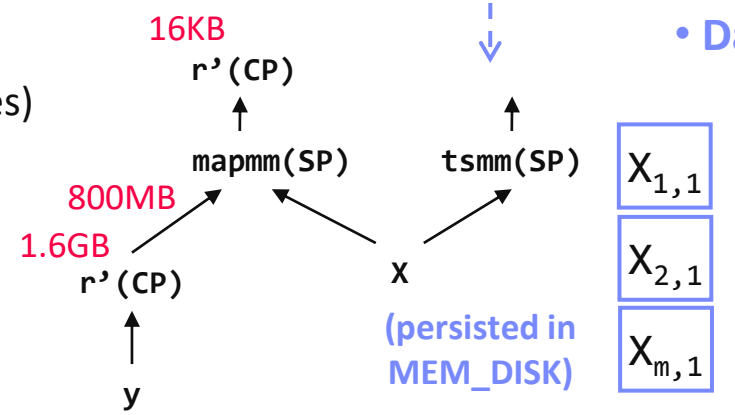
```
X = read($1);
y = read($2);
intercept = $3;
lambda = 0.001;
...
if( intercept == 1 ) {
  ones = matrix(1, nrow(X), 1);
  X = append(X, ones);
}
I = matrix(1, ncol(X), 1);
A = t(X) %*% X + diag(I)*lambda;
b = t(X) %*% y;
beta = solve(A, b);
...
write(beta, $4);
```

Scenario:
X: 10^8 x 10^3, 10^{11}
y: 10^8 x 1, 10^8

HOP DAG
(after rewrites)



LOP DAG
(after rewrites)



$X_{1,1}$
$X_{2,1}$
$X_{m,1}$

Cluster Config:
- driver mem: 20 GB
- exec mem: 60 GB

➔ **Distributed Matrices**
- Fixed-size matrix blocks
- Data-parallel operations

➔ **Hybrid Runtime Plans:**
- Size propagation / memory estimates
- Integrated CP / Spark runtime
- Dynamic recompilation during runtime

DAPHNE: An Open and Extensible System Infrastructure for Integrated Data Analysis Pipelines

https://github.com/daphne-eu/daphne

DAPHNE

BIFOLD

Motivation

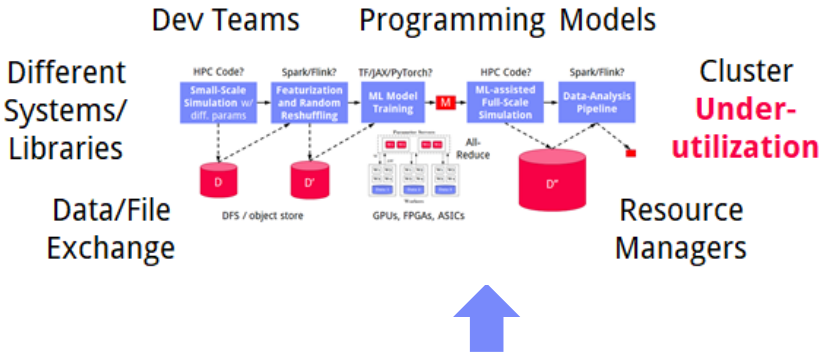
➔ DAPHNE Overall Objective: Open and extensible system infrastructure

- **Integrated Data Analysis Pipelines**
  - Open data formats, query processing
  - Data preprocessing and cleaning
  - ML model training and scoring
  - HPC, custom codes, and simulations

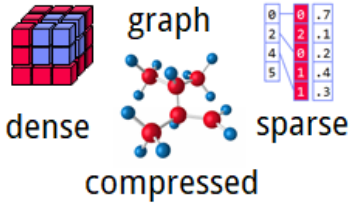
- **Hardware Challenges**
  - DM+ML+HPC share compilation and runtime techniques / converging cluster hardware
  - **End of Dennard scaling:** $P = \alpha CFV^2$ (power density 1)
  - **End of Moore's law**
  - **Amdahl's law:** sp = 1/s

  ➔ Increasing Specialization

Deployment Challenges



#1 Data Representations



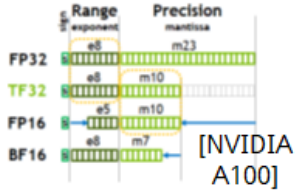
Sparsity Exploitation from Algorithms to HW

#2 Data Placement

Local vs **distributed**

#3 Data (Value) Types

FP32, FP64, INT8, INT32, INT64, UINT8, BF16, TF32, FlexPoint

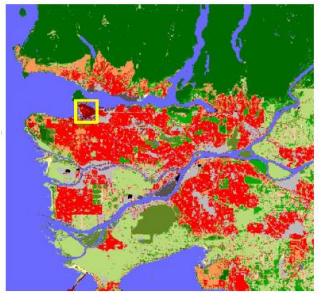


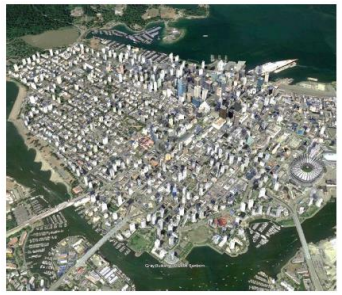
DAPHNE Use Cases

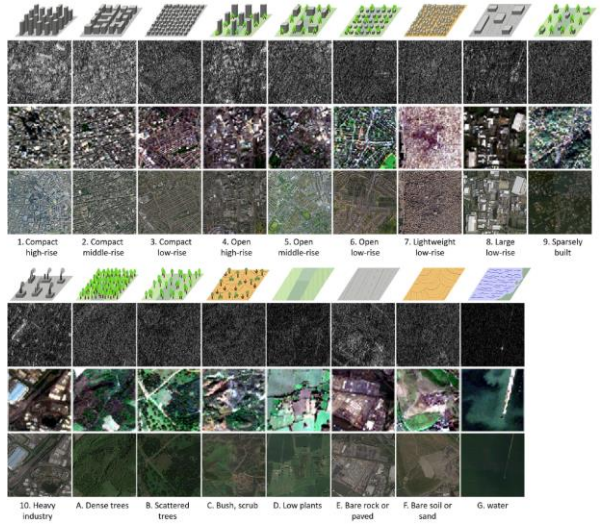
[Xiao Xiang Zhu et al: So2Sat LCZ42: A Benchmark Dataset for the Classification of Global Local Climate Zones. **GRSM 8(3) 2020**]

[So2Sat LC42: https://mediatum.ub.tum.de/1454690]

- **DLR Earth Observation**
  - **ESA Sentinel-1/2** datasets → 4PB/year
  - Training of local climate zone classifiers on **So2Sat LCZ42** (15 experts, 400K instances, 10 labels each, 85% confidence, ~55GB H5)
  - **ML pipeline:** preprocessing, ResNet20, climate models



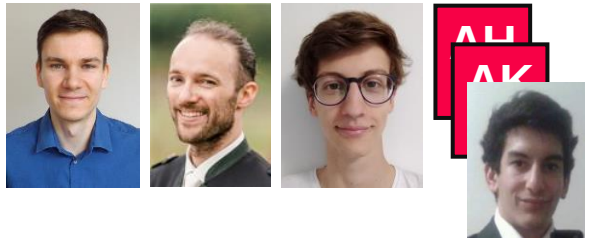


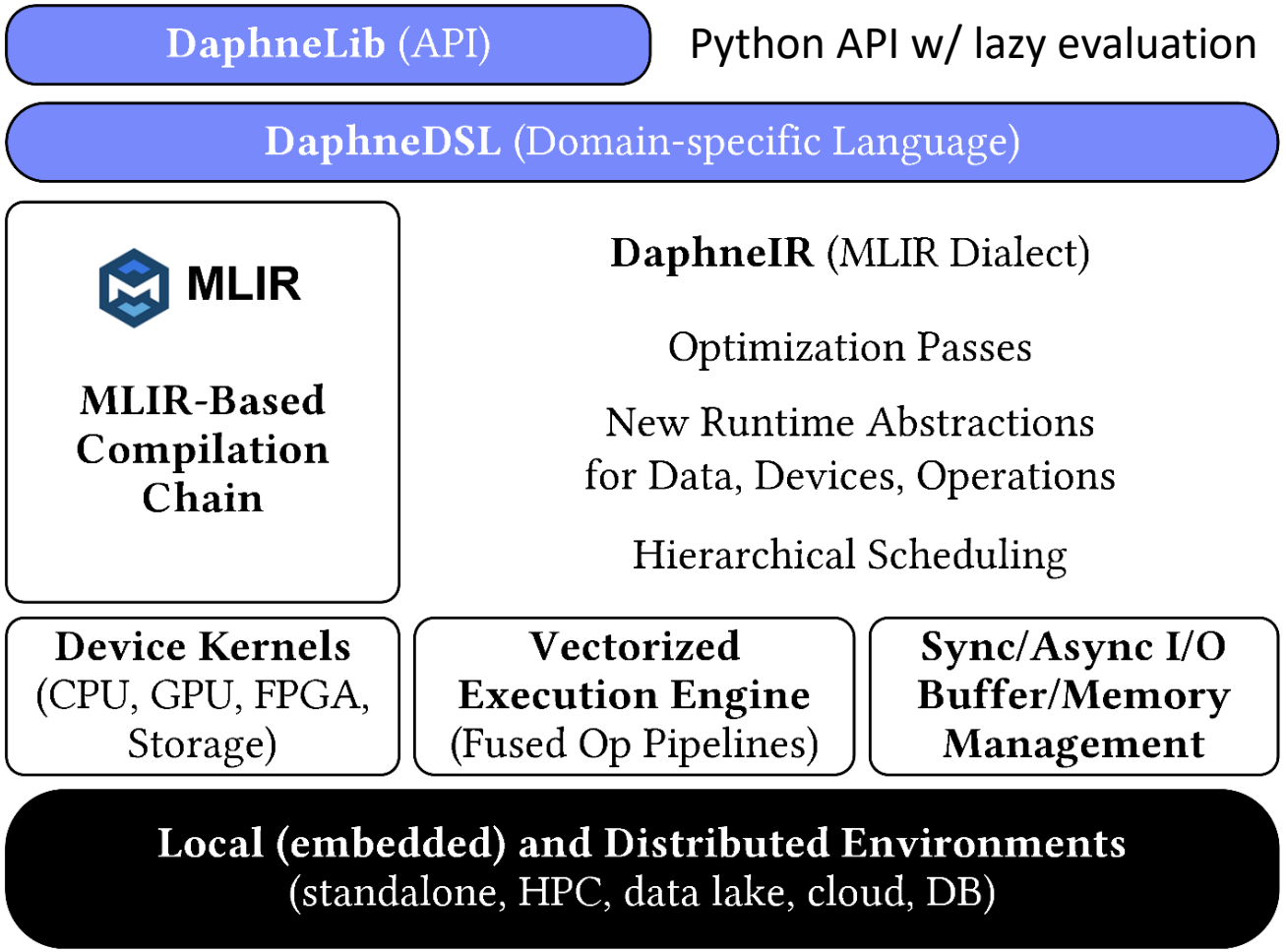


- **IFAT Semiconductor Ion Beam Tuning**
- **KAI Semiconductor Material Degradation**
- **AVL Vehicle Development Process (ejector geometries, KPIs)**

- **ML-assisted simulations, data cleaning, augmentation**

DAPHNE System Architecture [CIDR'22]





Extensible Infrastructure

MLIR Dialects, Extension Catalog (new data types, kernels, scheduling algs)

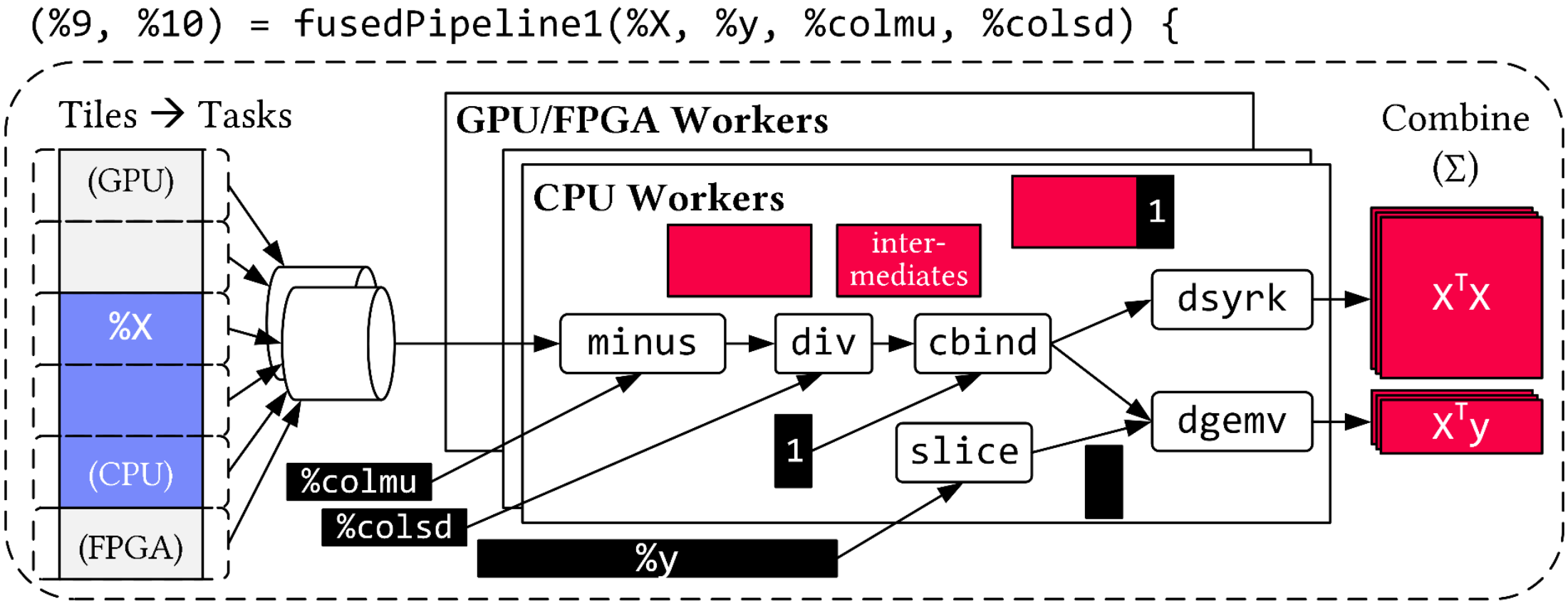
Multi-level Compilation/ Runtime

Sideways Entry, DSL-level constraints (e.g., data/op placement)

Fine-grained Fusion and Parallelism

Integration w/ Resource Mgmt & Prog. Models

Vectorized (Tiled) Execution



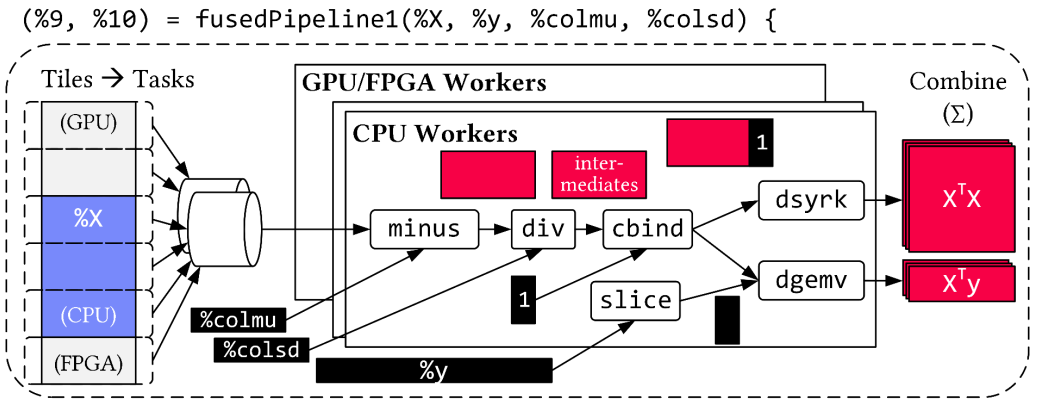
Default Parallelization
Frame & Matrix Ops

Locality-aware,
Multi-device Scheduling

Fused Operator Pipelines
on Tiles/Scalars + Codegen

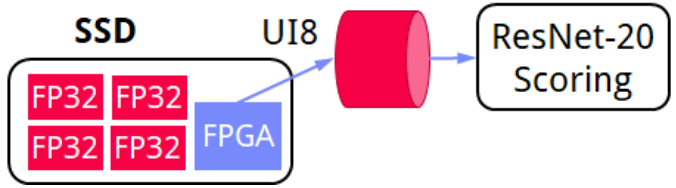
Vectorized (Tiled) Execution, cont.

- **#1 Zero-copy Input Slicing**
  - Create view on sliced input (no-op)
  - All kernels work on views
- **#2 Sparse Intermediates**
  - Reuse dense/sparse kernels
  - Sparse pipeline intermediates for free



- **#3 Fine-grained Control**
  - Task sizes (dequeue, data access) vs data binding (cache-conscious ops)
  - Scheduling for load balance (e.g., sparse operations)

- **#4 Computational Storage**
  - Task queues connect eBPF programs, async I/O into buffers, and op pipelines



List of Project Topics (Proposals)

Last update: Apr 17, 2023

#497 dml2daphnedsl: Translation of Domain-specific Languages

- **Motivation**
  - DAPHNE has a domain-specific language DaphneDSL for linear/relational algebra on matrices and frames, great for building complex data analysis algorithms
  - BUT: Data scientists prefer working at a higher abstraction level with primitives for common analytics tasks (e.g., data cleaning, clustering, …)
  - SystemDS has a powerful library of such primitives written in its domain-specific language DML
- **Task** (in Python or another suitable language)
  - Make SystemDS's library of ML primitives available to DAPHNE users
  - In that context: Stand-alone tool translating scripts written in SystemDS's DML to DAPHNE's DaphneDSL

Apache SystemDS™

```
shiftAndScale = function(Matrix[Double] X)
  return (Matrix[Double] R)
{
  R = (X - rowMeans(X)) / rowSds(X);
}
```

automatic translation

DAPHNE

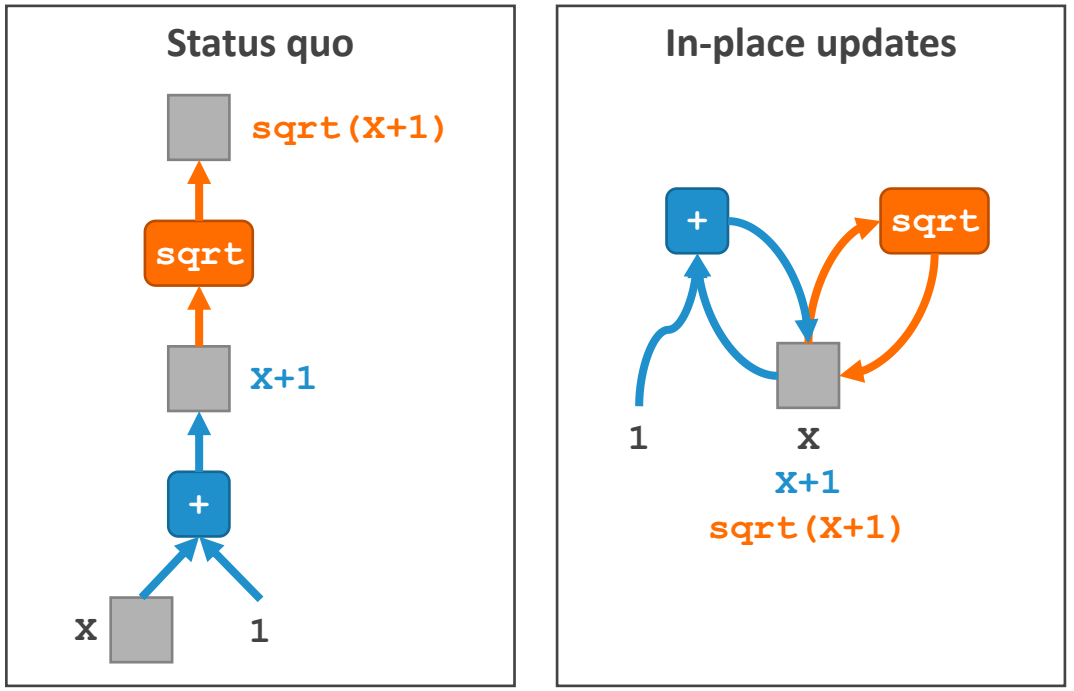
```
def shiftAndScale(X:matrix<f64>) -> matrix<f64> {
  return (X - mean(X, 1)) / stddev(X, 1);
}
```

- **More information & hints**
  - https://github.com/daphne-eu/daphne/issues/497

#498 Update-in-place Operations

- **Motivation**
  - DAPHNE maps logical ops from linear/relational algebra to calls to kernels (C++ functions consuming and producing DAPHNE matrices/frames)
  - Kernels always create a new data object for their output, causing additional memory accesses
  - *Under certain conditions*, the input data objects could be reused to improve memory consumption and performance
- **Task** (in C++)
  - Enable update-in-place optimizations in DAPHNE
  - Concerns DAPHNE compiler, runtime infra, kernels
- **More information & hints**
  - https://github.com/daphne-eu/daphne/issues/498

example: evaluation of `sqrt(X+1)` on matrix `X`

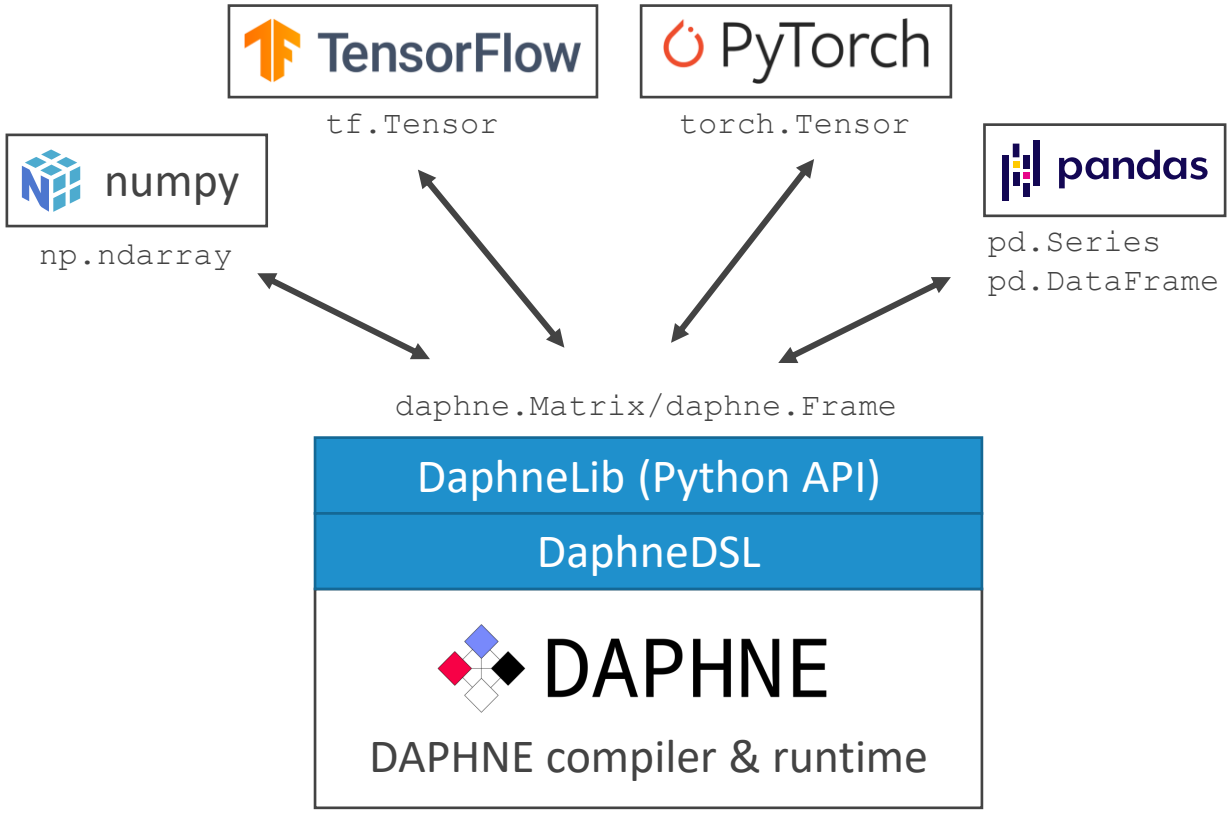


Newly created matrix after each operation

Reusing matrix X for both ops

#499 Connecting DAPHNE to the Data Science Ecosystem: Efficient Data Exchange with Popular Python Libs

- **Motivation**
  - Data scientists prefer Python nowadays
  - DAPHNE has a Python API (DaphneLib)
  - Efficient data exchange is required to successfully integrate DAPHNE with the data science ecosystem
- **Task** (in Python and C++)
  - Extend existing infra for efficient data exchange between DAPHNE and numpy by additional Python libraries such as pandas, TensorFlow, and PyTorch
- **More information & hints**
  - https://github.com/daphne-eu/daphne/issues/499

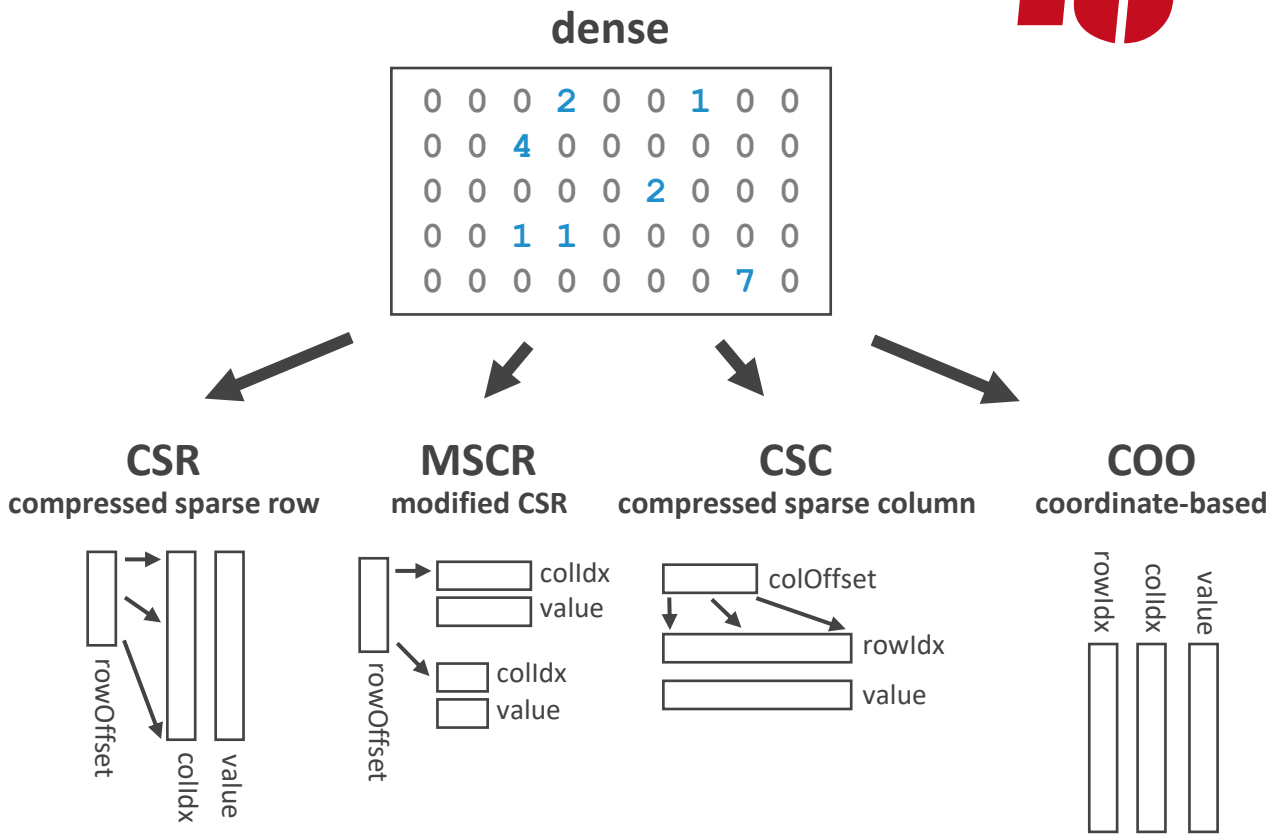


#500 End-to-end Sparsity Exploitation in DaphneDSL

- **Motivation**
  - Sparse matrices (most cells are zero) are commonplace in data science applications
  - DAPHNE supports a CSR (compressed sparse row) representation, but support is still a proof-of-concept
- **Task** (in C++)
  - Enable end-to-end sparsity exploitation (sparsity estimation, sparse kernels, dense/sparse selection, integration into DAPHNE's vectorized engine)
- **More information & hints**
  - https://github.com/daphne-eu/daphne/issues/500

#501 Additional Sparse Matrix Representations: MCSR, CSC, COO

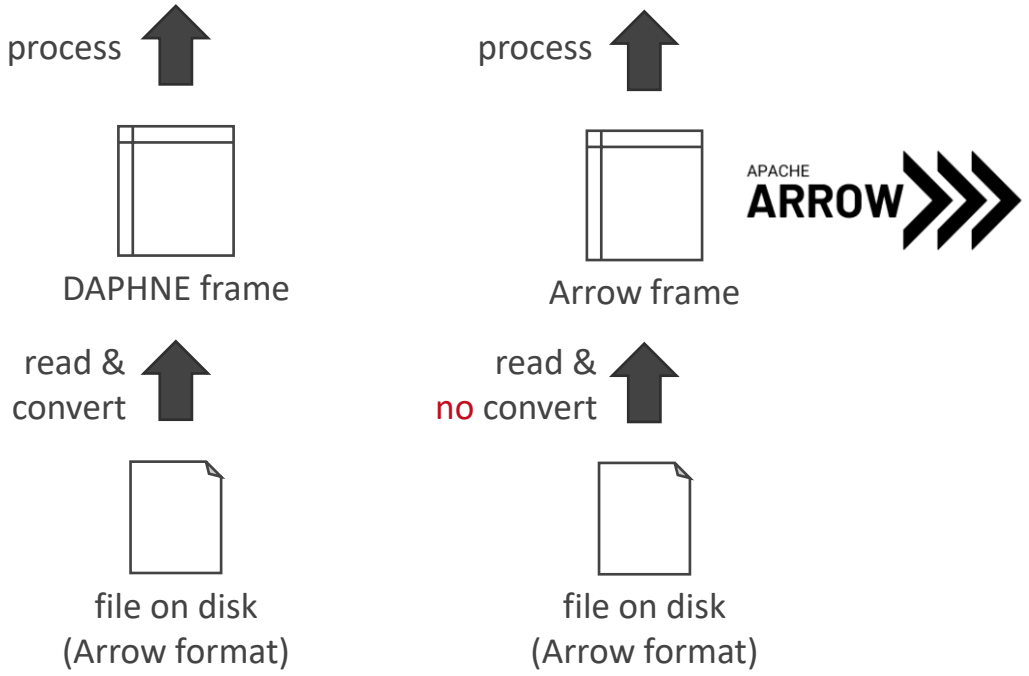
- **Motivation**
  - Sparse matrices (most cells zero) are commonplace in data science applications
  - Various sparse matrix representations with different characteristics, pros, and cons
  - So far, DAPHNE supports only CSR, would benefit from rich choice of representations to better adapt to various situations
- **Task** (in C++)
  - Extend DAPHNE by additional sparse matrix data types (e.g., MSCR, CSC, COO)
  - Add kernels (physical operators) with efficient algorithms specialized for these sparse formats
  - Devise selection of a suitable representation



- **More information & hints**
  - https://github.com/daphne-eu/daphne/issues/501

#502 Apache Arrow for Data Frames in DAPHNE

- **Motivation**
  - DAPHNE supports relational operations on frames, and offers a column-major frame representation
  - Apache Arrow is a widely used framework and exchange format for columnar data; supporting its memory layout in DAPHNE would improve the interoperability with other libraries and tools
  - In particular, reading Arrow files could become more efficient by avoiding unnecessary conversions
- **Task** (in C++)
  - Extend DAPHNE by a new frame data type which uses the memory layout of Apache Arrow
  - Add kernels for a few relational operations on the Arrow-based frame representation
  - Add a file reader for Arrow avoiding conversions



- **More information & hints**
  - https://github.com/daphne-eu/daphne/issues/502

More Topics in DAPHNE

- Improvements to Vectorized Engine and Operator Fusion
- DaphneDSL UDFs in Other Languages (e.g., Python, C++)
- Comprehensive Test Suite (Unit Tests and Script-Level Tests, DSL Fuzzing)

Graph Convolutional Layers in SystemDS

- **Motivation**
  - GCLs are the building blocks of many Graph Convolutional Networks (class of neural network for processing unstructured data representable as a graph)
  - Applications in social networks, molecular biology, multivariate timeseries
  - Key design element: pairwise message passing between neighbor nodes
  - Multiple types of mechanisms for pairwise message passing
- **Task** (in DML)
  - Implement the most relevant GCLs in SystemDS
  - Benchmark their performance on real-life data
- **More information & hints**
  - tba

DAG Visualizer for Machine Learning Workloads

- A typical machine learning (ML) system compiles a high-level script into hybrid execution plans of local, distributed and GPU operations. Understanding these plans are absolutely necessary to implement features in the compilation and runtime stacks. This project aims to build a toolkit that takes an execution plan of Apache SystemDS as input and produce a visual structure of the operations. A good example is TensorFlow's TensorBoard. However, unlike TensorBoard, this project aims to help the ML system researchers. The visualization may include various metadata associated with the operations such as predicted compute and memory costs, execution modes and instruction order. This visualization will allow the researchers to reason about alternative plans with multiple backends and devices. Furthermore, the toolkit should expose an API to add or remove metadata easily to support future work.

#3430: Query Interface over Lineage Traces

- **Task**
  - Build a query interface over many text-based lineage traces from various ML workloads. A lineage trace is a serialized DAG of the operations without any control flows. The task is to deserialize the traces into in-memory formats and answer queries regarding the workload characteristics. The in-memory format could be tabular or semi-structured. The internal representation should preserve the structure of the DAGs and the operators' properties to answer all kinds of queries. One possible way would be to represent each DAG by multiple relations – one for each operator with the corresponding attributes, and one for preserving the structure with attributes including output nodes for each operator. Existing libraries (e.g., Pandas) can be used to define the query interface. Example queries include: Find all DAGs with a convolution operation that takes more than 20ms to execute. Compare the total number of operations between two DAGs. Group DAGs by the type of non-linear operation used and calculate the average execution time for each group. Compare the memory usage of the matrix multiplication between two DAGs. Find similar DAGs on different datasets.
- **More information & hints**
  - https://issues.apache.org/jira/browse/SYSTEMDS-3430

#3151/#3163: Missing Value Imputation & Amputation

- **Motivation**
  - Missing data values are a typical issue encountered in real-world data sets
- **Task**
  - Given a data set with missing values
    - Classify the missing values as Missing Completely at Random (MCAR), Not Missing at Random (NMAR), and Missing not at Random (MNAR)
    - If NMAR: Impute the missing values
  - Given a dataset without missing values: ampute a part of the values with various techniques
- **More information & hints**
  - https://issues.apache.org/jira/browse/SYSTEMDS-3151
  - https://issues.apache.org/jira/browse/SYSTEMDS-3163

#3178: Tuple Deduplication

- **Motivation**
  - Deduplication is an important data cleaning/preprocessing step
- **Task** (in DML)
  - Implement a dedup() built-in function in SystemDS
  - Challenge: Make pairwise comparison efficient (e.g., clustering/blocking, hashing, sorting)
- **More information & hints**
  - https://issues.apache.org/jira/browse/SYSTEMDS-3178

More Topics in Apache SystemDS

- **Alternative Linear Algebra Kernels**
  - E.g., transposition-aware matrix multiplications
  - E.g., Intel OneAPI instead of Intel MKL
- **Extended Matrix Multiplication Chain Optimization**
- **Federated/Distributed Frame Operations**
- **Additional GPU Kernels (e.g., remove empty rows, cumsum)**
- **Auto Differentiation**
- **Loop Vectorization**
- **Extended Common Subexpression Elimination**
- **Extended I/O Framework: Readers/Writers for More File Formats (NetCDF, HDF5, Arrow)**

Alternativ: 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
  - Other open-source systems possible, but contributions might be more difficult to get accepted

Summary and Q&A

- FG Big Data Engineering (DAMS Lab)
- Course Organization, Outline, and Deliverables
- Apache SystemDS
- DAPHNE
- List of Project Topics (Proposals)

- Remaining Questions?

- See you during the office hours ☺