SCIENCE
PASSION
TECHNOLOGY

# Architecture of ML Systems
# 01 Introduction and Overview

**Matthias Boehm**

Graz University of Technology, Austria
Computer Science and Biomedical Engineering
Institute of Interactive Systems and Data Science
BMK endowed chair for Data Management

Last update: Mar 04, 2021

# Announcements/Org

- **#1 Video Recording**
  - Link in TeachCenter & TUbe (lectures will be public)
  - Optional attendance (independent of COVID)
  - Hybrid, in-person but video-recorded lectures
    - **RED**: webex https://tugraz.webex.com/meet/m.boehm
    - **ORANGE** (Mar 15): in-person in i5 w/ TUbe video recording

- **#2 Course Registrations** (as of Mar 04)
  - **Architecture of Machine Learning Systems** (AMLS): **106 (9)**
  - Bachelor/master/PhD ratio?

- **#3 Siemens Student Challenge**
  - ML model for classification w/ dependability assessment
  - Submission deadline: **May 02**, total prices: **10.000 EUR**

[https://ecosystem.siemens.com/ai-da-sc]

# Agenda

- **Data Management Group**
- **Motivation and Goals**
- **Course Organization**
- **Course Outline, and Projects**
- **Overview Apache SystemDS**

# Data Management Group

https://damslab.github.io/

# About Me

- **09/2018 TU Graz**, Austria
  - BMK endowed chair for data management
  - **Data management for data science**
    (ML systems internals, end-to-end data science lifecycle)

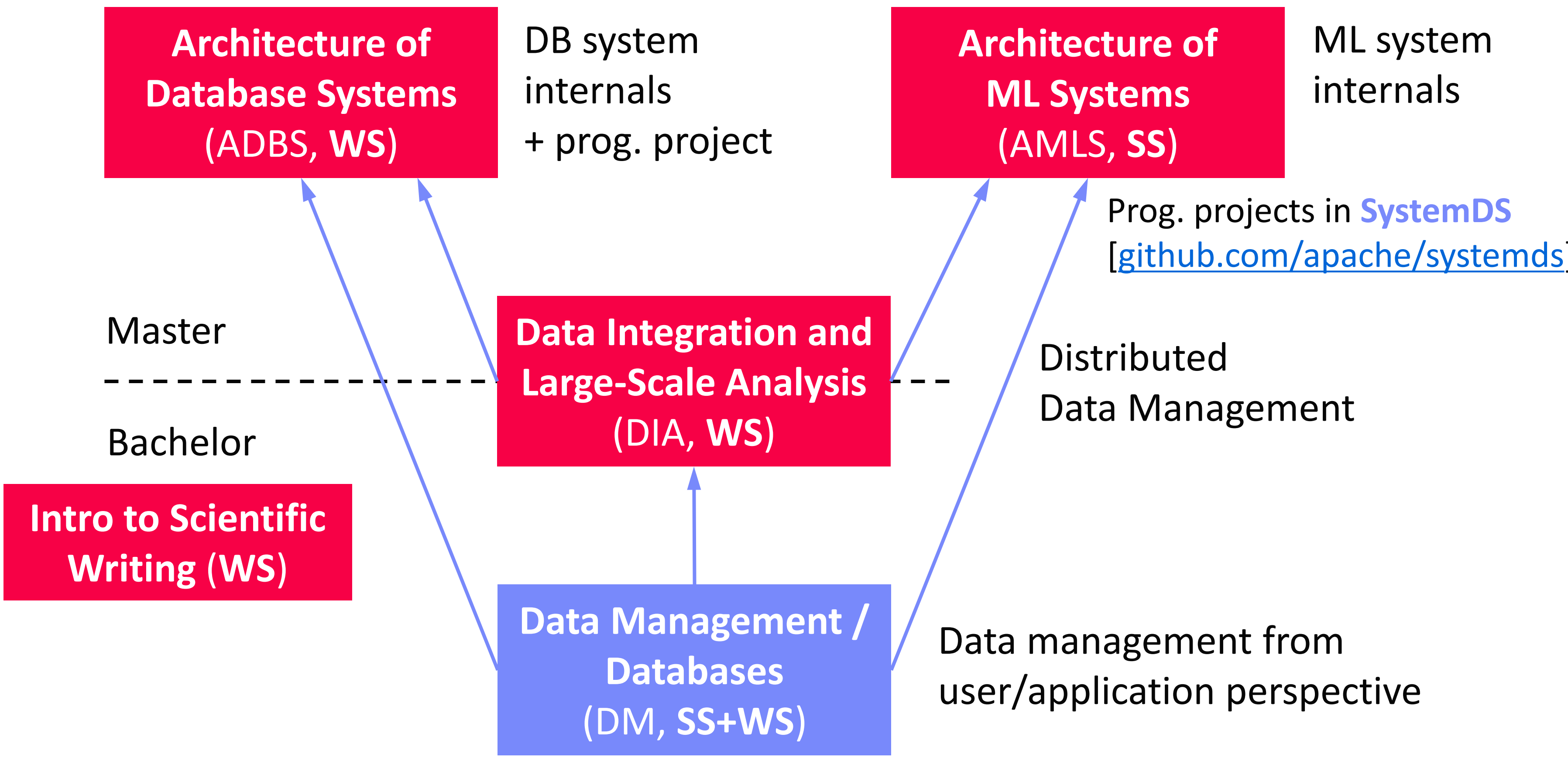
https://github.com/apache/systemds

- **2012-2018 IBM Research – Almaden**, USA
  - Declarative large-scale machine learning
  - Optimizer and runtime of **Apache SystemML**

- **2011 PhD TU Dresden**, Germany
  - Cost-based optimization of integration flows
  - Systems support for time series forecasting
  - In-memory indexing and query processing

DB group

# Data Management Courses

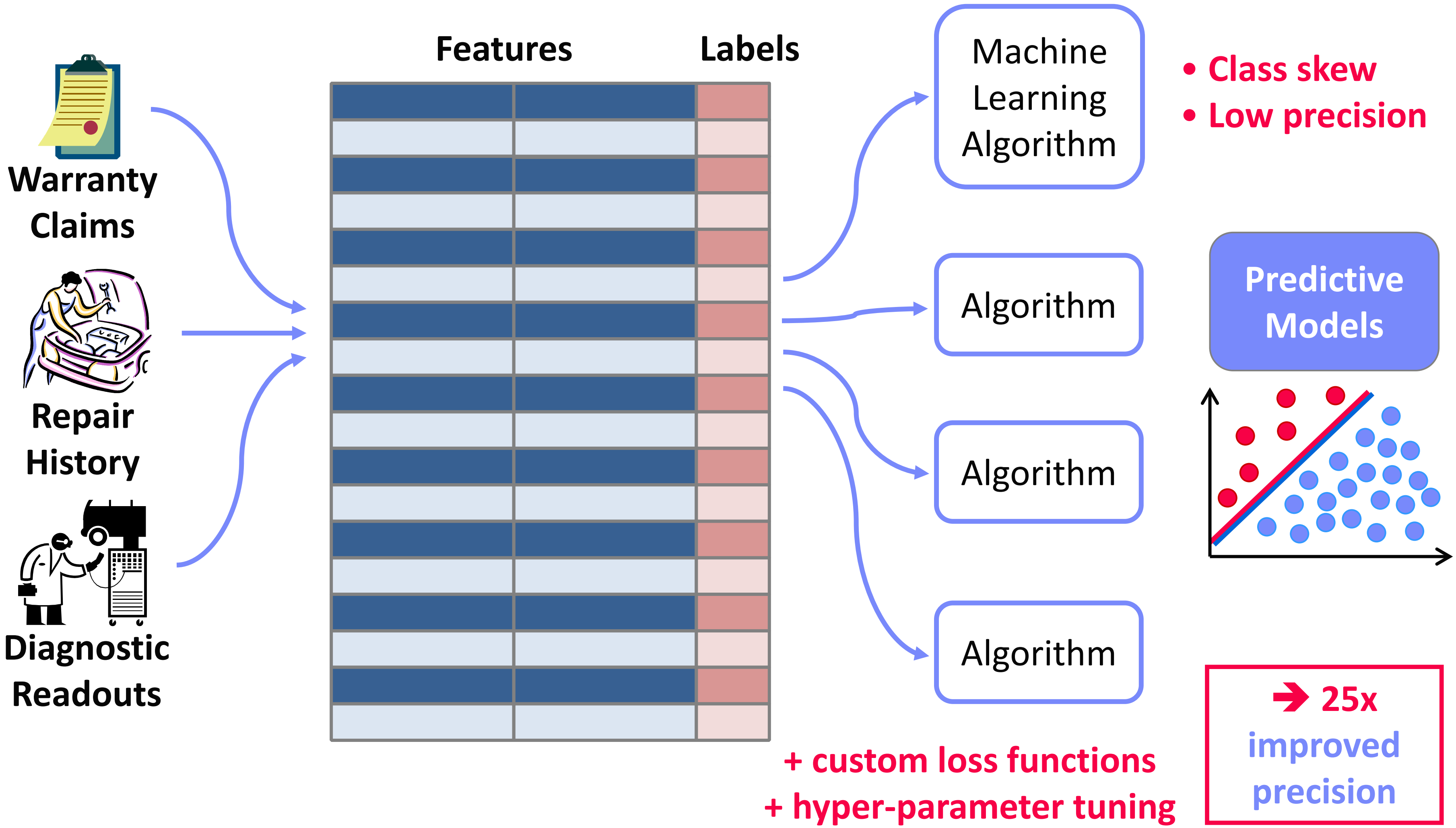


# Motivation and Goals

# Example ML Applications (Past/Present)

- **Transportation / Space**
  - **Lemon car detection and reacquisition** (classification, seq. mining)
  - **Airport passenger flows from WiFi data** (time series forecasting)
  - **Data analysis for assisted driving** (various use cases)
  - **Automotive vehicle development** (ML-assisted simulations)
  - Satellite senor analytics (regression and correlation)
  - Earth observation and **local climate zone classification** and monitoring
- **Finance**
  - Water cost index based on various influencing factors (regression)
  - **Insurance claim cost per customer** (model selection, regression)
  - **Financial analysts survey correlation** (bivariate stats w/ new tests)
- **Health Care**
  - **Breast cancer cell grow from histopathology images** (classification)
  - **Glucose trends and warnings** (clustering, classification)
  - Emergency room diagnosis / patient similarity (classification, clustering)
  - Patient survival analysis and prediction (Cox regression, Kaplan-Meier)

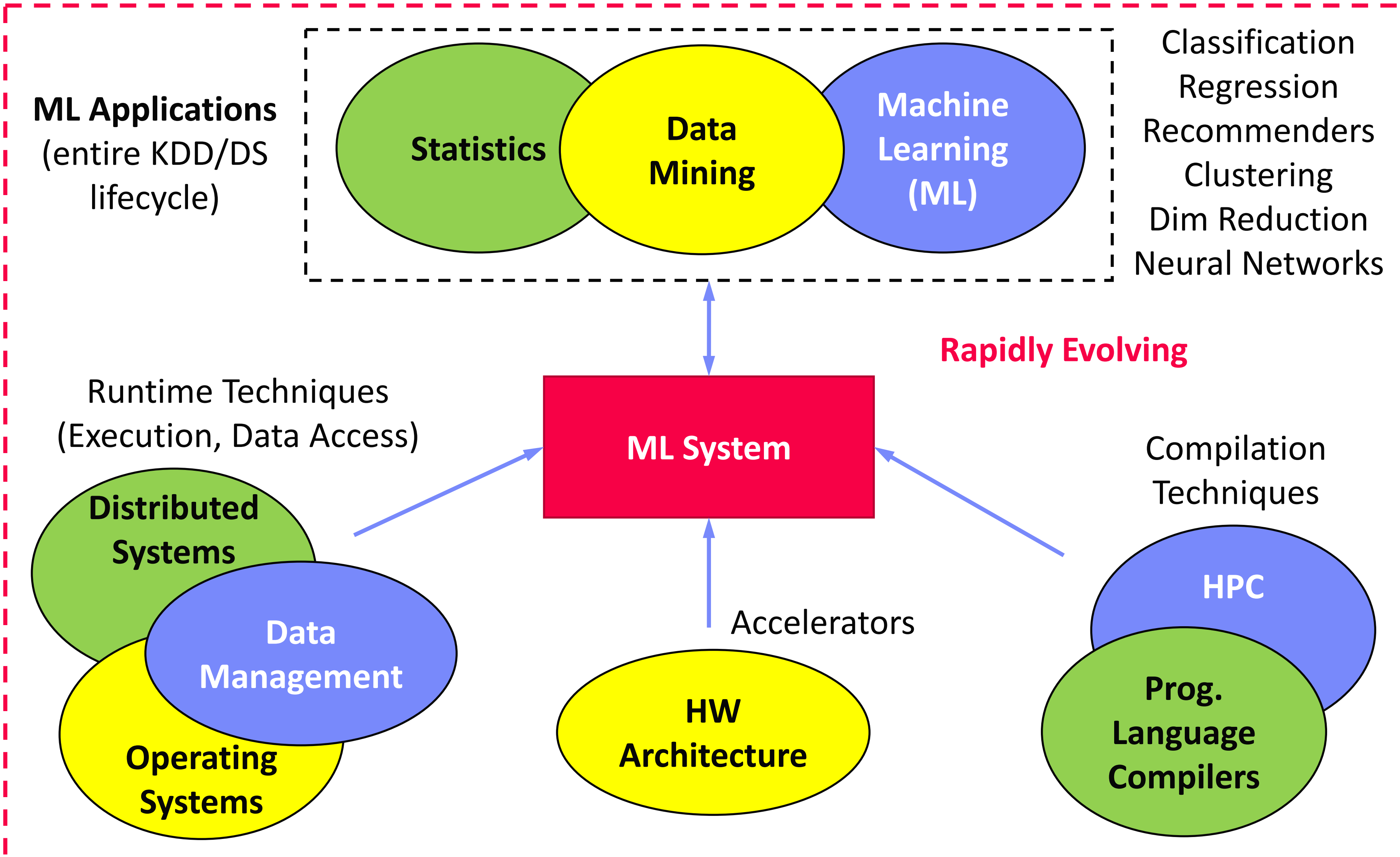
# A Car Reacquisition Scenario



# Example ML Applications (Past/Present), cont.

- **Production/Manufacturing**
  - **Paper and fertilizer production** (regression/classification, anomalies)
  - **Semiconductor manufacturing**, and **material degradation** modeling
- **Other Domains**
  - **Machine data: errors and correlation** (bivariate stats, seq. mining)
  - Smart grid: energy demand/RES supply, weather models (forecasting)
  - Visualization: dimensionality reduction into 2D (auto encoder)
  - **Elastic flattening via sparse linear algebra** (spring-mass system)
- **Information Extraction**
  - **NLP contracts → rights/obligations** (classification, error analysis)
  - **PDF table recognition and extraction, OCR** (NMF clustering, custom)
  - **Learning explainable linguistic expressions** (learned FOL rules, classification)
- **Algorithm Research** (+ various state-of-the art algorithms)
  - **User/product recommendations** via various forms of NMF
  - Localized, supervised metric learning (dim reduction and classification)
  - Learning word embeddings via orthogonalized skip-gram

# What is an ML System?



# What is an ML System?, cont.

- **ML System**
  - **Narrow focus:** SW system that executes ML applications
  - **Broad focus:** Entire system (HW, compiler/runtime, ML application)
  - → Trade-off **runtime/resources** vs **accuracy**
  - → Early days: no standardizations (except some exchange formats), lots of different languages and system architectures, but many shared concepts

- **Course Objectives**
  - Architecture and internals of modern (large-scale) ML systems
    - **Microscopic view** of ML system internals
    - **Macroscopic view** of ML pipelines and data science lifecycle
  - **#1** Understanding of characteristics → **better evaluation / usage**
  - **#2** Understanding of effective techniques → **build/extend ML systems**

# Course Organization

# Basic Course Organization

- **Staff**
  - Lecturer: Univ.-Prof. Dr.-Ing. Matthias Boehm, ISDS
  - Assistant: M.Sc. Sebastian Baunsgaard, ISDS
- **Language**
  - Lectures and slides: **English**
  - Communication and examination: **English**/**German**
- **Course Format**
  - VU 2/1, **5 ECTS** (2x 1.5 ECTS + 1x 2 ECTS), bachelor/master
  - **Weekly lectures** (**start 12.15pm**, including **Q&A**), **attendance optional**
  - **Mandatory programming project** (2 ECTS)
  - **Recommended papers** for additional reading on your own
- **Prerequisites (preferred)**
  - Basic courses Data Management/Databases, and
  - Basic courses on applied ML / Knowledge Discovery and Data Mining

# Course Logistics

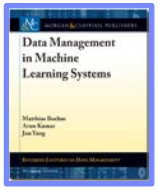
- **Website**
  - https://mboehm7.github.io/teaching/ss21_amls/index.htm
  - All course material (lecture slides) and dates
- **Video Recording Lectures (TUbe, webex)?**
- **Communication**
  - **Informal language** (first name is fine)
  - Please, **immediate feedback** (unclear content, missing background)
  - Newsgroup: N/A – email is fine, summarized in following lectures
  - **Office hours:** by appointment or after lecture
- **Exam**
  - **Completed programming project** (checked by me/staff), **~June 30**
  - **Final written exam** (oral exam if <=25 students take the exam)
  - **Grading** (40% project/exercises completion, 60% exam)

# Course Logistics, cont.

- **Course Applicability**
  - **Master** programs computer science (CS), as well as software engineering and management (SEM)
    - Catalog Data Science (compulsory course in major, and elective)
    - Catalog Machine Learning (elective course)
    - Catalog Interactive and Visual Information Systems (elective course)
    - Catalog Software Technology (elective course)
  - **PhD** CS doctoral school list of courses
  - **Free subject course** in any other study program or university

# Course Outline and Projects

Partially based on




[Matthias Boehm, Arun Kumar, Jun Yang: Data Management in Machine Learning Systems. Synthesis Lectures on Data Management, Morgan & Claypool Publishers 2019]

**Major updates in SS2020 and SS2021**

# Part A: Overview and ML System Internals

- **01 Introduction and Overview** [Mar 05]
- **02 Languages, Architectures, and System Landscape** [Mar 12]

- **03 Size Inference, Rewrites, and Operator Selection** [Mar 19]
- **04 Operator Fusion and Runtime Adaptation** [Mar 26]
- **05 Data- and Task-Parallel Execution** [Apr 16]
- **06 Parameter Servers** [Apr 23]

- **07 Hybrid Execution and HW Accelerators** [Apr 30]
- **08 Caching, Partitioning, Indexing, and Compression** [Apr 07]

# Part B: ML Lifecycle Systems

- **09 Data Acquisition, Cleaning, and Preparation** [May 21]
- **10 Model Selection and Management** [May 28]
- **11 Model Debugging, Fairness, and Explainability** [Jun 04]
- **12 Model Serving Systems and Techniques** [Jun 11]

- **13 Q&A and Exam Preparation**

# Programming Projects

- **Open Source Projects**
  - Programming project in context of open source projects
    - **Apache SystemDS:** https://github.com/apache/systemds
    - **DAPHNE:** https://daphne-eu.github.io/
      (private repo but OSS release ~01/2022)
    - Other OSS projects possible, but harder to merge PRs
  - Commitment to **open source and open communication** (PRs, mailing list)
  - **Remark:** Don't be afraid to ask questions / develop code in public

- **Objectives**
  - Non-trivial feature in an ML system (**2 ECTS → 50 hours**)
  - **OSS processes:** Break down into subtasks, code/tests/docs, PR per project, code review, incorporate review comments, etc
- **Team**
  - Individuals or up to **three-person teams** (w/ separated responsibilities)

# Programming Projects, cont.

- **Alternative Exercise: Siemens Student Challenge**
  - ML model for classification w/ dependability assessment
  - (Submission deadline: **May 02**, total prices: **10.000 EUR**)

SIEMENS

[https://ecosystem.siemens.com/ai-da-sc]

- **Task: Develop an ML model that classifies given datasets and provides explanations for the misclassification probability**
  - Each team receives three labeled datasets A, B, C (csv files), generated from a chosen probability distribution on a subset of $[0,1]^2$
  - Traffic light labels (red/green)
    - **False red** prediction → cost but no safety problem
    - **False green** prediction → safety problem
  - Classifier and non-trivial upper-bounds for misclassification probability
  - Up to **three-person teams** (university students w/o completed PhD)
  - Paper on the proposed approach (up to 10 A4 pages, >=10pt font)
    - Including assumptions, and extension proposal for n-dim

SCIENCE
PASSION
TECHNOLOGY

# Apache SystemDS: An ML System for the End-to-End Data Science Lifecycle

**Matthias Boehm**[1,2], **Iulian Antonov**[2], **Sebastian Baunsgaard**[1], **Mark Dokter**[2], **Robert Ginthör**[2], **Kevin Innerebner**[1], **Florijan Klezin**[2], **Stefanie Lindstaedt**[1,2], **Arnab Phani**[1], **Benjamin Rath**[1], **Berthold Reinwald**[3], **Shafaq Siddiqi**[1], **Sebastian Benjamin Wrede**[2]

[1] **Graz University of Technology**; Graz, Austria
[2] **Know-Center GmbH**; Graz, Austria
[3] **IBM Research – Almaden**; San Jose, CA, USA

# 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"**

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

**[NIPS 2015]**

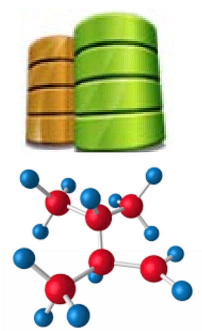
**[DEBull 2018]**

# The Data Science Lifecycle

**Data-centric View:**
Application perspective
Workload perspective
System perspective

Data extraction, schema alignment, entity resolution, data validation, data cleaning, outlier detection, missing value imputation, semantic type detection, data augmentation, feature selection, feature engineering, feature transformations

Data Scientist



**Data Integration**
**Data Cleaning**
**Data Preparation**

**Model Selection**
**Training**
**Hyper-parameters**

**Validate & Debug**
**Deployment**
**Scoring & Feedback**

**Exploratory Process**
(experimentation, refinements, ML pipelines)

Data/SW Engineer

DevOps Engineer

**Key observation: SotA data integration/cleaning based on ML**

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

Background and System Architecture

https://github.com/apache/systemds

# Example: Linear Regression Conjugate Gradient

**Note:**
**#1 Data Independence**
**#2 Implementation-Agnostic Operations**

```
X = read($1); # n x m matrix
y = read($2); # n x 1 vector
maxi = 50; lambda = 0.001;
intercept = $3;
...
r = -(t(X) %*% y);
norm_r2 = sum(r * r); p = -r;
w = matrix(0, ncol(X), 1); i = 0;
while(i<maxi & norm_r2>norm_r2_trgt)
{
   q = (t(X) %*% (X %*% p))+lambda*p;
   alpha = norm_r2 / sum(p * q);
   w = w + alpha * p;
   old_norm_r2 = norm_r2;
   r = r + alpha * q;
   norm_r2 = sum(r * r);
   beta = norm_r2 / old_norm_r2;
   p = -r + beta * p; i = i + 1;
}
write(w, $4, format="text");
```

Read matrices from HDFS/S3

Compute initial gradient

Compute conjugate gradient

Compute step size

Update model and residuals

➔ **"Separation of Concerns"**

# Apache SystemML/SystemDS

**APIs:** Command line, JMLC, Spark MLContext, Spark ML, (20+ Scalable Algorithms)

DML Scripts

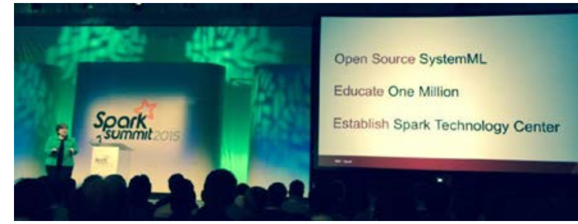
**Language**

**Compiler**

**Runtime**

Write Once, Run Anywhere

**[SIGMOD'15,'17,'19]**
**[PVLDB'14,'16a,'16b,'18]**
**[ICDE'11,'12,'15]**
**[CIDR'17]**
**[VLDBJ'18]**
**[DEBull'14]**
**[PPoPP'15]**




Apache SystemML™

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

**In-Memory Single Node**
(scale-up)

**Hadoop or Spark Cluster**
(scale-out)

**In-Progress:**

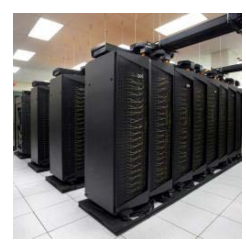
GPU

**since 2014/16**

Java

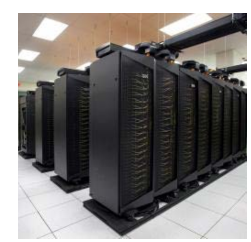
**since 2012**

hadoop



**since 2010/11**

Apache Spark



**since 2015**

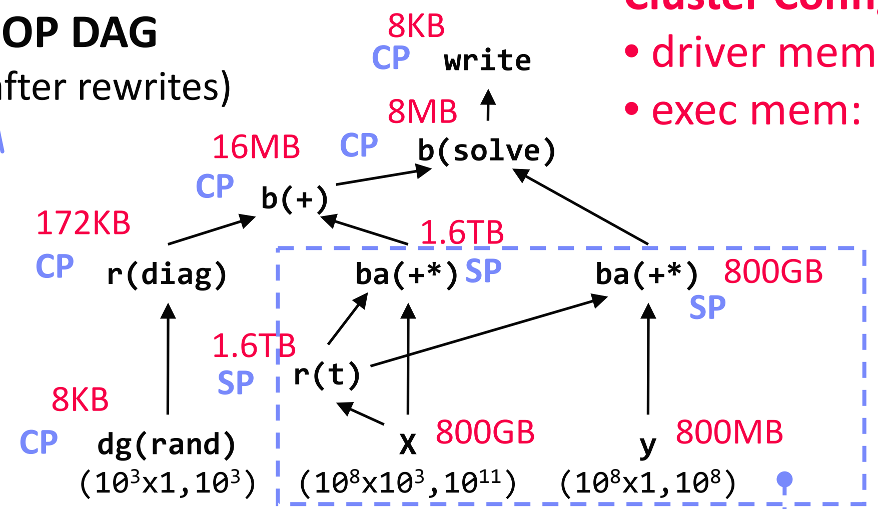
# Basic HOP and LOP DAG Compilation

## 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);
```

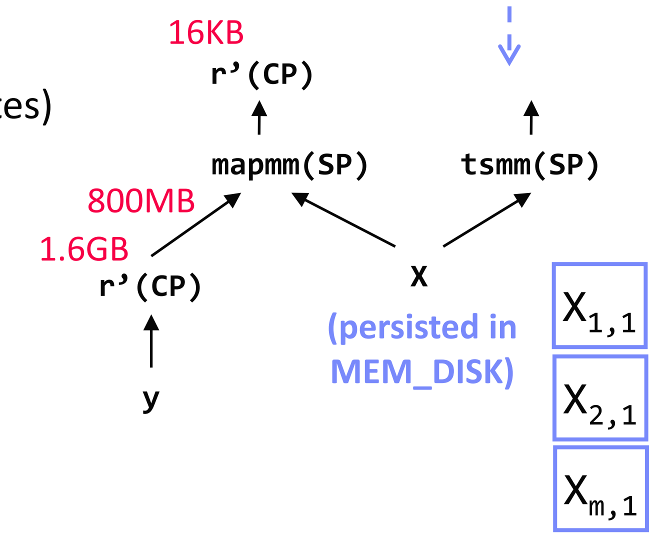
**Scenario:**
X: $10^8$ x $10^3$, $10^{11}$
y: $10^8$ x 1, $10^8$

**HOP DAG**
(after rewrites)




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

**LOP DAG**
(after rewrites)



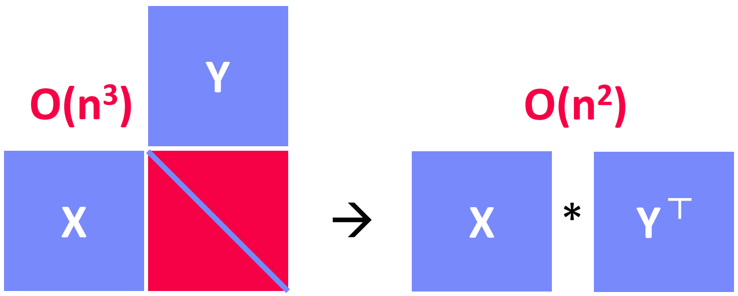

➔ **Hybrid Runtime Plans:**
- Size propagation / memory estimates
- Integrated CP / Spark runtime
- Dynamic recompilation during runtime

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

# Static and Dynamic Rewrites

- **Example Static Rewrites** (size-indep.)
  - Common Subexpression Elimination
  - Constant Folding / Branch Removal / Block Sequence Merge
  - Static Simplification Rewrites
  - Right/Left Indexing Vectorization
  - For Loop Vectorization
  - Spark checkpoint/repartition injection

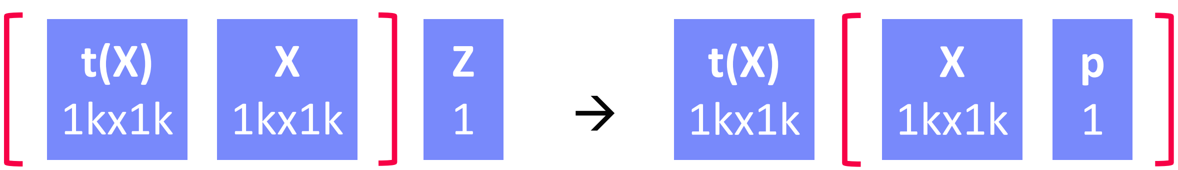
```
trace(X%*%Y) → sum(X*t(Y))
```




```
sum(λ*X) → λ*sum(X)
sum(X+Y) → sum(X)+sum(Y)
```

- **Example Dynamic Rewrites** (size-dep.)
  - Dynamic Simplification Rewrites
  - Matrix Mult Chain Optimization

```
rowSums(X) → X, iff ncol(X)=1
sum(X^2) → X%*%t(X), iff ncol(X)=1
```

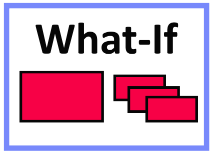



2,002 MFLOPs

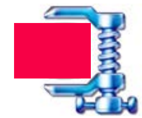
4 MFLOPs

**Size propagation and sparsity estimation**

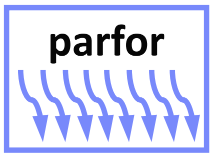
# Selected Research Results




**#3 Resource Optimization**
for automatic resource provisioning
(SIGMOD'15)



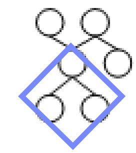
**#4 Compressed Linear Algebra**
(PVLDB'16,
SIGMOD Record'17,
VLDB Journal'18, CACM'19)




**#2 Task-Parallel Parfor Loops**
hybrid parallelization strategies
(PVLDB'14)

Apache SystemML™

(ICDE'11,
PVLDB'16)



**#5 Optimizing Operator Fusion Plans**
(PPoPP'15, CIDR'17,
PVLDB'18)

**#1 SystemML's Optimizer**
rewrites, operator selection, size propagation, memory estimates, dynamic recompilation (DEBull'14)

$\Sigma\Pi$

**#6 Advanced Optimization**
sum-product (CIDR'17),
sparsity estimation (SIGMOD'19)

Google Summer of Code

GPU, meta, numerical stability, parameter servers, etc

# Lessons Learned from SystemML

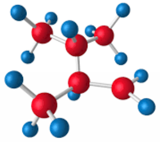
**Why was SystemML not adopted in practice?**

- **L1 Data Independence & Logical Operations**
  - Independence of **evolving technology stack** (MR → Spark, GPUs)
  - **Simplifies development** (libs) and **deployment** (large-scale vs. embedded)
  - **Enables adaptation** to cluster/data characteristics (dense/spare/compressed)

- **L2 User Categories** (|Alg. Users| **>>** |Alg. Developers|)
  - **Focus on ML researchers** and algorithm developers **is a niche**
  - Data scientists and domain experts **need higher-level abstractions**

- **L3 Diversity of ML Algorithms & Apps**
  - **Variety of algorithms** (batch 1st/2nd, mini-batch DNNs, hybrid)
  - Different parallelization, ML + rules, numerical computing

- **L4 Heterogeneous Structured Data**
  - Support for **feature transformations on 2D frames**
  - Many apps deal with **heterogeneous data and various structure**

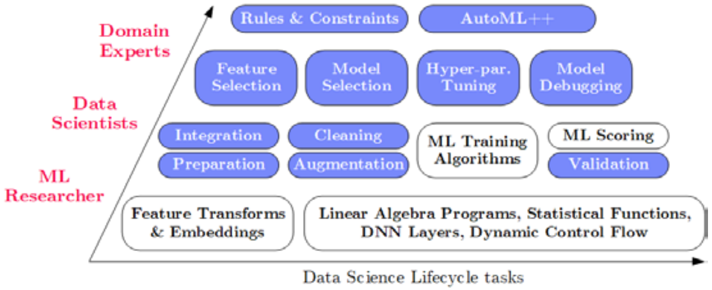


# Apache SystemDS Design

Apache SystemML (since 2010)
→ SystemDS (09/2018)
→ Apache SystemDS (07/2020)

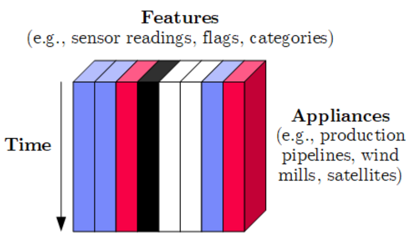
- **Objectives**
  - Effective and efficient **data preparation, ML, and model debugging at scale**
  - High-level abstractions for different lifecycle tasks and users

- **#1 Based on DSL for ML Training/Scoring**
  - Hierarchy of **abstractions for DS tasks**
  - ML-based SotA, interleaved, performance




- **#2 Hybrid Runtime Plans and Optimizing Compiler**
  - System infrastructure for diversity of algorithm classes
  - Different parallelization strategies and new architectures (**Federated ML**)
  - Abstractions → redundancy → automatic optimization

- **#3 Data Model: Heterogeneous Tensors**
  - Data integration/prep requires **generic data model**



# Language Abstractions and APIs, cont.

- **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 = ... }
}
```

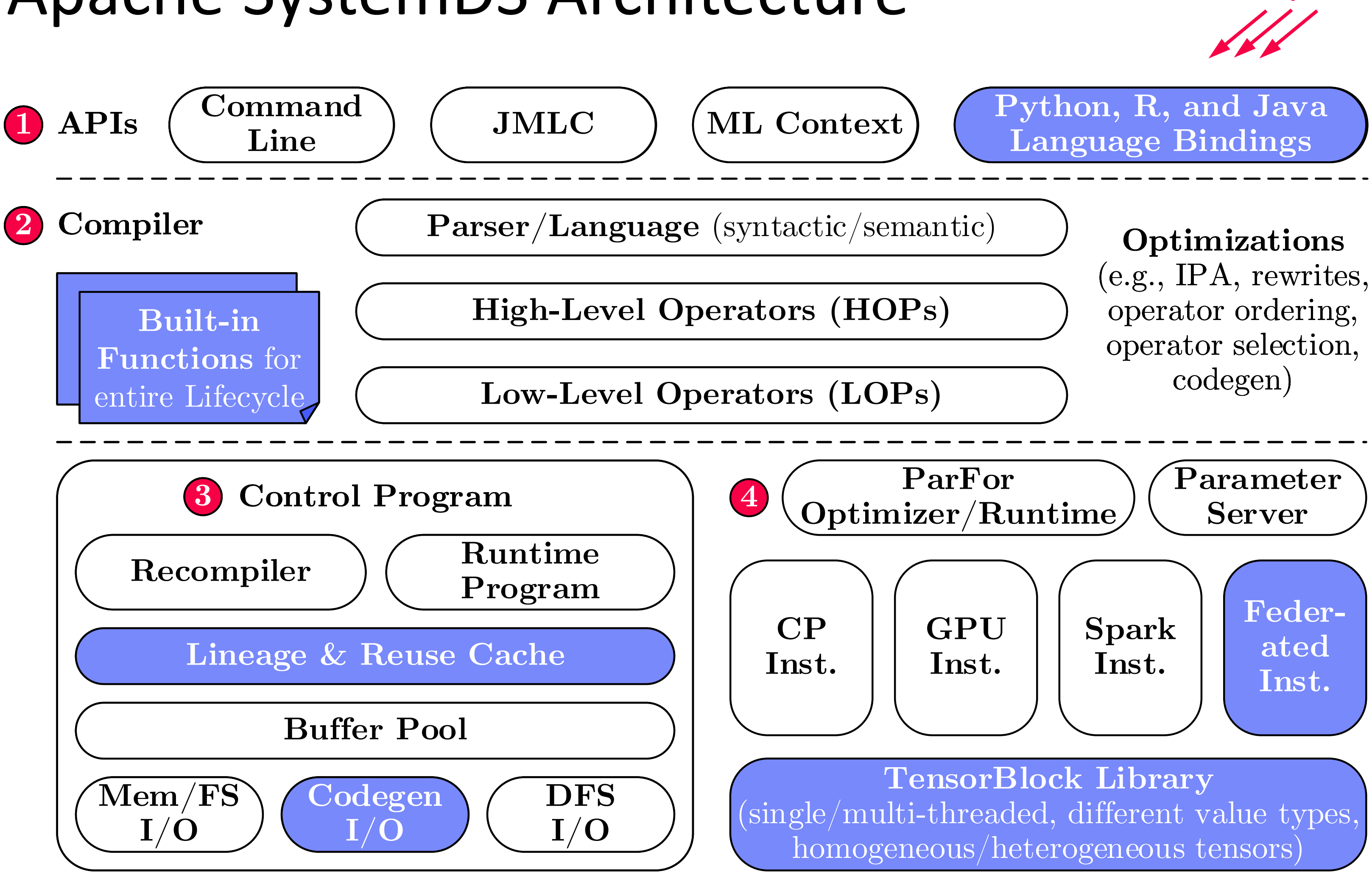
```
m_lmDS = function(...) {
  l = matrix(reg,ncol(X),1)
  A = t(X) %*% X + diag(l)
  b = t(X) %*% y
  beta = solve(A, b) ...}
```

Linear Algebra Programs

**Facilitates optimization across data science lifecycle tasks**

# Apache SystemDS Architecture

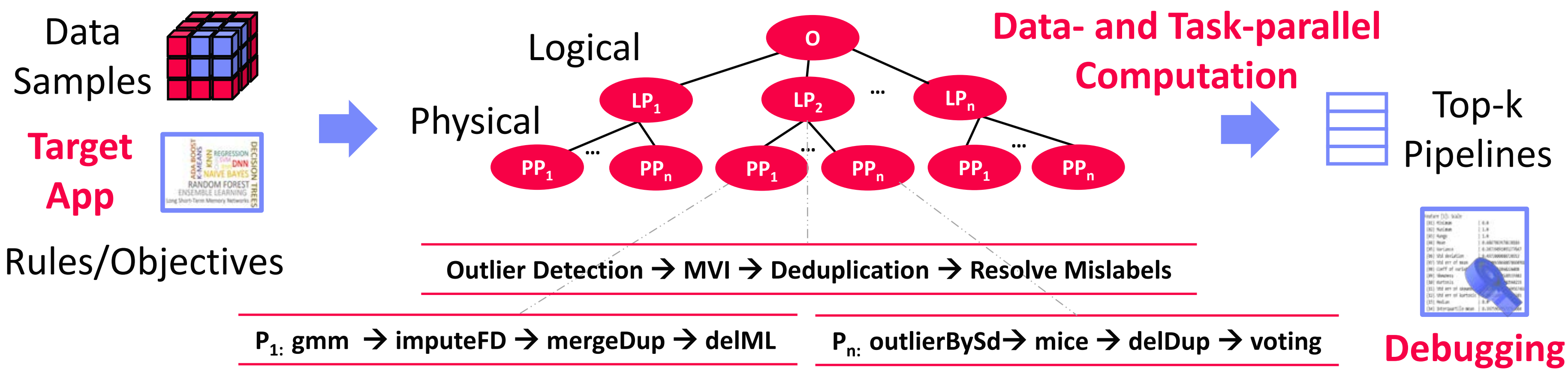
> **17,500** tests




[M. Boehm, I. Antonov, S. Baunsgaard, M. Dokter, R. Ginthör, K. Innerebner, F. Klezin, S. N. Lindstaedt, A. Phani, B. Rath, B. Reinwald, S. Siddiqui, S. Benjamin Wrede: SystemDS: A Declarative Machine Learning System for the End-to-End Data Science Lifecycle. **CIDR 2020**]

# Data Cleaning Pipelines

- **Automatic Generation of Cleaning Pipelines**
  - Library of robust, parameterized **data cleaning primitives** (physical/logical)
  - **Enumeration of DAGs** of primitives & **hyper-parameter optimization** (HB, BO)




| University | Country |
|---|---|
| TU Graz | Austria |
| TU Graz | Austria |
| TU Graz | Germany |
| IIT | India |
| IIT | IIT |
| IIT | Pakistan |
| IIT | India |
| SIBA | Pakistan |
| SIBA | null |
| SIBA | null |

**Dirty Data**

| University | Country |
|---|---|
| TU Graz | Austria |
| TU Graz | Austria |
| TU Graz | Austria |
| IIT | India |
| IIT | India |
| IIT | India |
| IIT | India |
| SIBA | Pakistan |
| SIBA | Pakistan |
| SIBA | Pakistan |

After **imputeFD(0.5)**

| A | B | C | D |
|---|---|---|---|
| 0.77 | 0.80 | 1 | 1 |
| 0.96 | 0.12 | 1 | 1 |
| 0.66 | 0.09 | null | 1 |
| 0.23 | 0.04 | 17 | 1 |
| 0.91 | 0.02 | 17 | null |
| 0.21 | 0.38 | 17 | 1 |
| 0.31 | null | 17 | 1 |
| 0.75 | 0.21 | 20 | 1 |
| null | null | 20 | 1 |
| 0.19 | 0.61 | 20 | 1 |
| 0.64 | 0.31 | 20 | 1 |

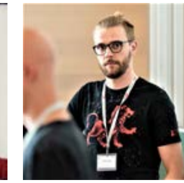
**Dirty Data**

| A | B | C | D |
|---|---|---|---|
| 0.77 | 0.80 | 1 | 1 |
| 0.96 | 0.12 | 1 | 1 |
| 0.66 | 0.09 | 17 | 1 |
| 0.23 | 0.04 | 17 | 1 |
| 0.91 | 0.02 | 17 | 1 |
| 0.21 | 0.38 | 17 | 1 |
| 0.31 | 0.29 | 17 | 1 |
| 0.75 | 0.21 | 20 | 1 |
| 0.41 | 0.24 | 20 | 1 |
| 0.19 | 0.61 | 20 | 1 |
| 0.64 | 0.31 | 20 | 1 |

After **MICE**

# Multi-Level Lineage Tracing & Reuse

[SIGMOD'21]



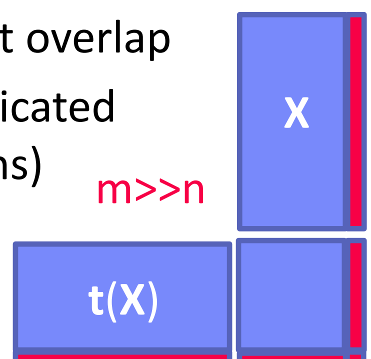
- **Lineage as Key Enabling Technique**
  - Trace lineage of operations (incl. non-determinism), dedup for loops/functions
  - Model versioning, data reuse, incremental maintenance, autodiff, debugging

- **Full Reuse of Intermediates**
  - Before executing instruction, probe output lineage in cache `Map<Lineage, MatrixBlock>`
  - Cost-based/heuristic caching and eviction decisions (compiler-assisted)

```
for( i in 1:numModels )
  R[,i] = lm(X, y, lambda[i,], ...)
```

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

- **Partial Reuse of Intermediates**
  - **Problem:** Often partial result overlap
  - Reuse partial results via dedicated rewrites (compensation plans)
  - Example: steplm




```
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)
} }
```

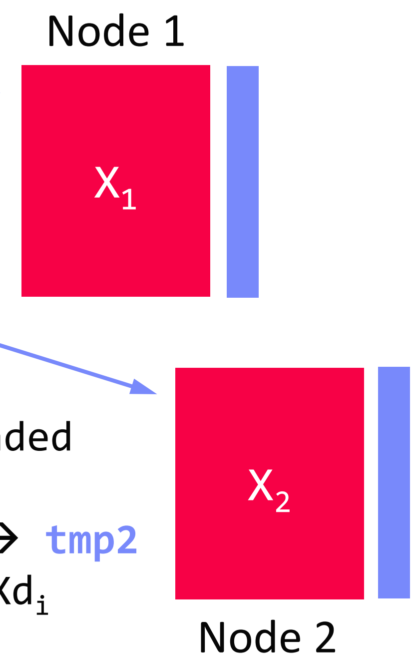
# Federated Learning [SIGMOD'21]

- **Python API**
  - Federated data objects and lazy evaluation

```
features = federated(sds,[node1,node2],([…],[…]))
model = features.l2svm(labels).compute()
```

- **Example Federated Execution**

```
while(continueOuter & iter<maxi) {
  Xd = X %*% s (federated MV)
  # ...
  while(continueInner) {
    out = 1-Y* (Xw+step_sz*Xd);
    sv = (out > 0);
    out = out * sv;
    g = wd + step_sz*dd
         - sum(out * Y * Xd);
    h = dd + sum(Xd * sv * Xd);
    step_sz = step_sz - g/h;
  }
  g_new = t(X) %*% (out * Y)
         - lambda * w
  # ...
} ...
```



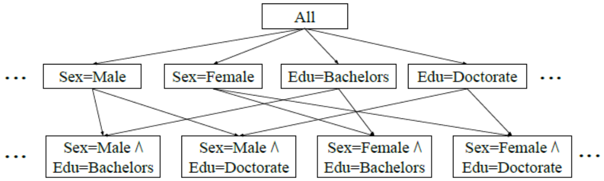

```
# At all workers
0. load Xi if not loaded
1. Send s → tmp1
2. Exec Xi %*% tmp1 → tmp2
3. Retrieve tmp2 as Xdi

# At master
Xd = rbind(Xd1, Xd2)
```

# Model Debugging

- **Problem:** Model M with 85% accuracy
  - **Find top-k data slices where model performs worse** than average
  - Data slice: $S^{DG}$ := D=PhD ∧ G=female (subsets of features)
  - Score: $\mathbf{w} * \text{err}(S^{DG})/\text{err}(S^{*}) + (\mathbf{1-w}) * |S^{DG}|$

[Yeounoh Chung et al.: Slice Finder: Automated Data Slicing for Model Validation. CoRR 2018/**ICDE2019**]

- **Existing Algorithms**
  - Binning + One-Hot Encoding of **X**
  - Lattice search w/ heuristic, level-wise termination




- **Extensions**
  - **#1 Lower/upper bounds sizes/errors** → pruning & termination
  - **#2 Scalable implementation in linear algebra** (join & eval via sparse-sparse matrix multiply)

Candidate Slices

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Data

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 2 & 0 \\ 0 & 2 & 0 \\ 2 & 0 & 1 \\ 0 & 2 & 0 \\ 1 & 1 & 1 \\ 2 & 0 & 1 \end{bmatrix}$$

**== Level**

# Summary & Q&A

# Thanks

- Data Management Group
- Motivation and Goals
- Course Organization
- Course Outline, and Projects
- Overview Apache SystemDS

→ **Programming Projects** in **Apache SystemDS**, **DAPHNE**, other OSS ML Systems, or **Siemens Student Challenge**

- **Next Lectures**
  - **02 Languages, Architectures, and System Landscape** [Mar 12] **+ project topics**
  - **03 Size Inference, Rewrites, and Operator Selection** [Mar 19]
  - **04 Operator Fusion and Runtime Adaptation** [Mar 26]
  - **05 Data- and Task-Parallel Execution** [Apr 16]
  - **06 Parameter Servers** [Apr 23]
  - **07 Hybrid Execution and HW Accelerators** [Apr 30]
  - **08 Caching, Partitioning, Indexing and Compression** [May 07]