

Introduction to Scientific Writing

01 Structure of Scientific Papers

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: Oct 29, 2020

Announcements/Org

- **#1 Virtual Lectures**

- <https://tugraz.webex.com/meet/m.boehm>
- Optional attendance (independent of COVID)



- **#2 Course Registrations** (as of Oct 28)

- Changes in WS20/21
- **Introduction to Scientific Writing**

ISDS Group

Boehm

20

Agenda

- **Data Management Group**
- **Course Organization, Outline, and Projects**
- **Structure of Scientific Papers**

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](https://github.com/apache/systemds)

- **2012-2018 IBM Research – Almaden, USA**

- Declarative large-scale machine learning
- Optimizer and runtime of **Apache SystemML**



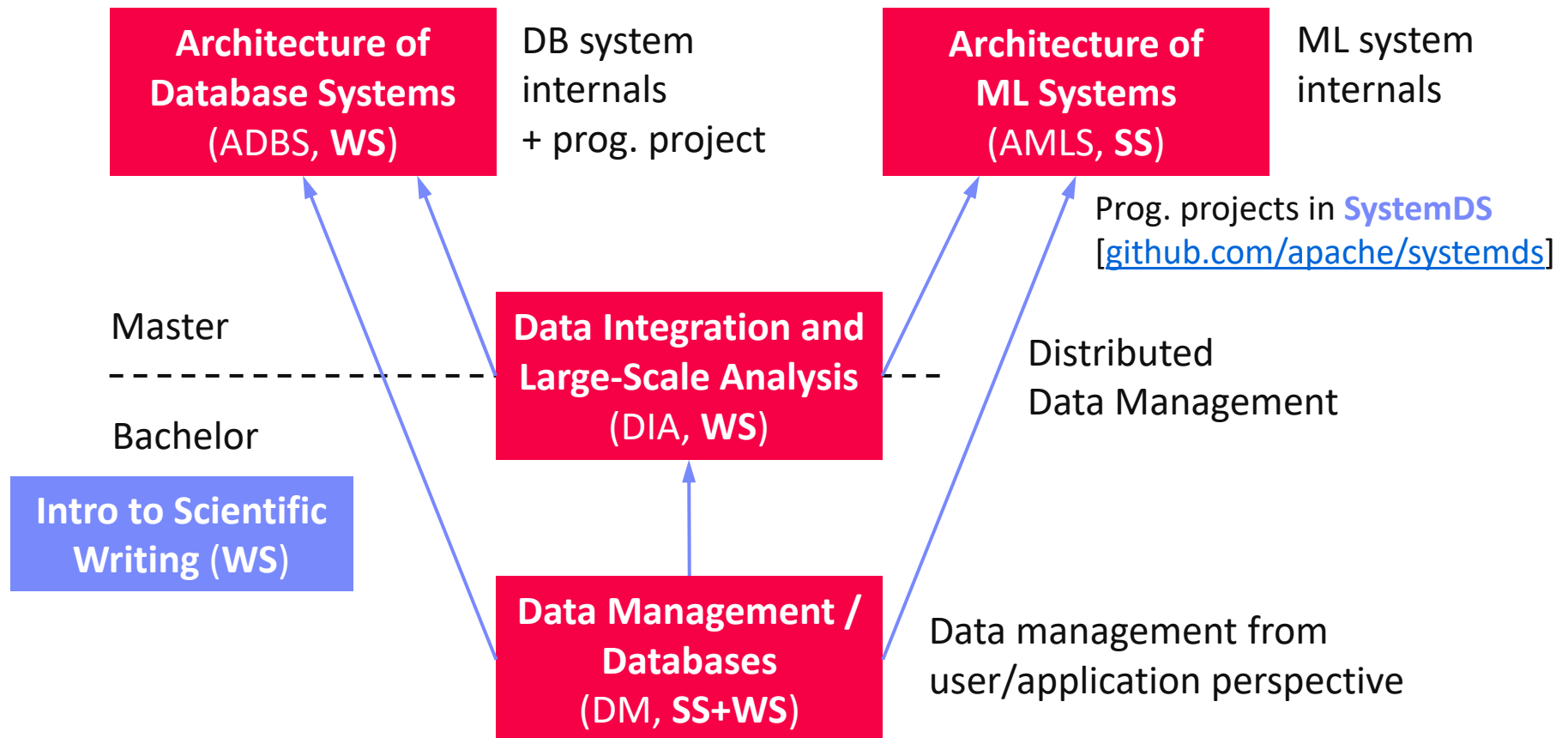
- **2011 PhD TU Dresden, Germany**

- Cost-based optimization of integration flows
- Systems support for time series forecasting
- In-memory indexing and query processing



DB group

Data Management Courses



Course Organization, Outline, Goals, and Projects

Course Logistics

■ Staff

- Lecturer: Univ.-Prof. Dr.-Ing. Matthias Boehm, ISDS
- Assistant: M.Sc. Shafaq Siddiqi, ISDS



■ Language

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

■ Course Format

- SE 1, **2 ECTS** (0.5 ECTS lectures + 1.5 ECTS paper/talk), bachelor
- 3 lectures, **optional attendance**
- Mandatory paper and presentation

Course Logistics, cont.

■ Website

- https://mboehm7.github.io/teaching/ws2021_isw/index.htm
- All course material (lecture slides) and dates

■ Video Recording Lectures (T**U**be)? **No**

- Lectures and discussions via Webex
- No recording in order to foster discussion, private presentations



■ Goals

- Understanding of / communication through scientific writing
- Best practices for effective scientific reading, writing, and reproducibility

■ Grading

- **Overall:** **pass**/**fail** (no detailed 1-5 grades)
- Includes submitted paper and final presentation
(pass := adhere to given constraints + acceptable quality)

Outline Lectures

- **01 Structure of Scientific Papers** [Oct 29, 6pm, **optional**]
- **02 Scientific Reading and Writing** [Nov 05, 6pm, **optional**]
- **03 Experiments, Reproducibility, and Projects** [Nov 12, 6pm, **optional**]
- ...
- **04 Project Presentations** [Jan 07, 6pm, **mandatory**]

Paper Projects

Alternative: LV combined
with bachelor thesis

■ Team

- 1-4 person teams (w/ clearly separated responsibilities)

■ Project

- Pick from a given list of papers / groups of papers
- **#1** Write short summary paper (#pages = 2 * team-size,
written in LaTeX, ACM acmart template, document-class sigconf, PDF)
- **#2** Prepare and present talk on paper summary (7min + 3min Q&A)

■ Timeline

- **Nov 12:** List of projects proposals, feel free to bring your own
- **Dec 23:** paper submission via email to m.boehm@tugraz.at (11.59pm)
- **Jan 07:** Final project presentation (all students)

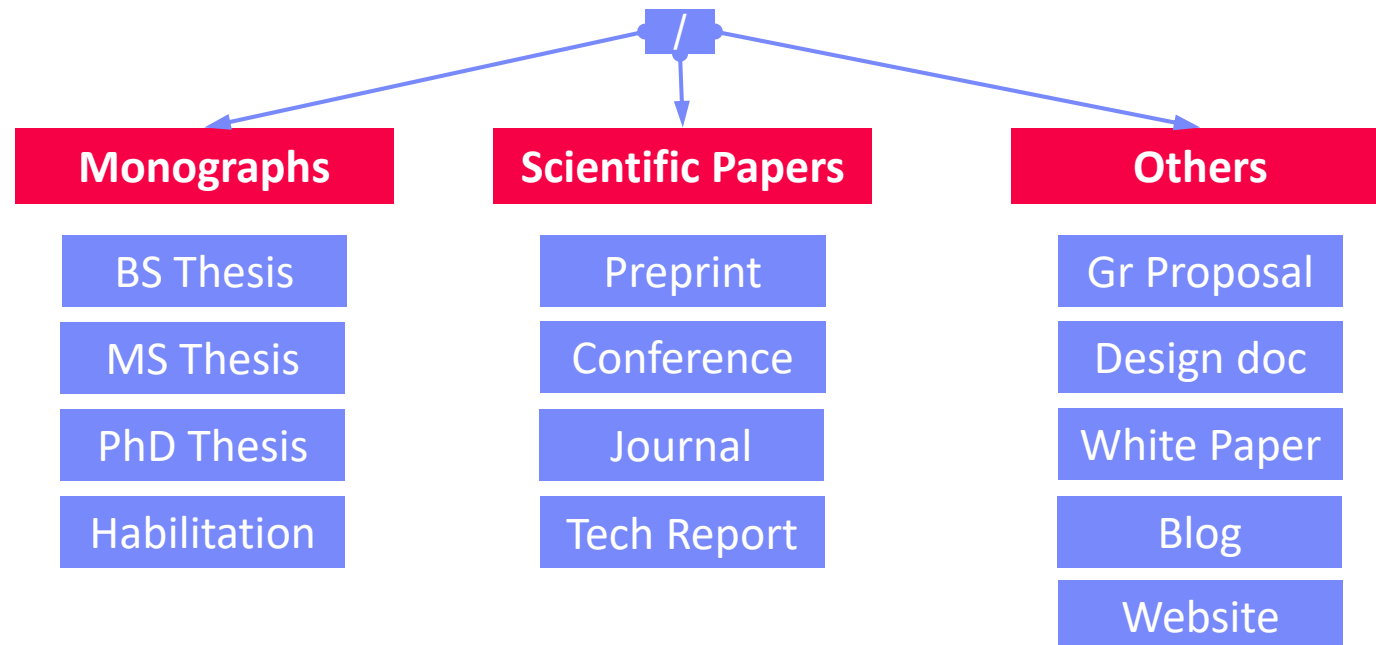
Structure of Scientific Papers

In Computer Science (Data Management)

Overview Types of Scientific Writing

■ Classification of Scientific/Technical Documents

- Formal vs informal writing, cumulative?, single vs multi-author
- Archival vs non-archival publications



➔ Scientific Writing Skills are **crucial**

- Different types of docs share many similarities

Preparation

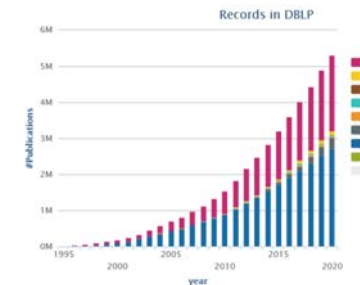
■ #1 Know your Audience

■ #2 Get your Workflow in Order

- Writing: LaTeX (e.g., Overleaf, TeXnicCenter), versioning (e.g., git), templates
- Plotting: R (plot, ggplot), Python (matplotlib), Gnuplot
- Figures: MS Visio, Inkscape → pdf, eps, svg (vector graphics)

■ #3 Mindset: Quality over Quantity

- Aim for top-tier conferences/journals (act as filter)
- Make the paper useful for others (ideas, evidence, code)
- **Example** (my own theses/books)
 - Seminar (~bachelor), 5 months, **446** pages
 - Diploma (~master), 9 months, **274** pages
 - PhD thesis, 4 years, **237** pages
 - 1st book, 5+2 years, **157** pages



Your reader's time is
a scarce resource

Paper Writing and Publication Process

■ Research – Writing Cycle

- Read lots of papers
- ~~Idea~~ → ~~Research~~ → ~~Writing~~ → ~~Document~~
- **Idea** → Writing/Research → Document
- Incremental refinement of drafts

■ Paper Submission Cycle

- Blind vs double-blind submission
- Revisions and Camera-ready
- **Similar: bachelor/master thesis**
→ drafts to advisor / final version

■ Demo CMT

- Example SIGMOD 2021



[Recommended Reading]

[Simon Peyton Jones: How to write a great research paper, MSR Cambridge]



[Eamonn Keogh: How to do good research, get it published in SIGKDD and get it cited!, **KDD 2009**]



Example SIGMOD 2020

October 15, 2019: Abstract submission

October 22, 2019: Paper submission

December 10 - 11, 2019: Author responses

January 17, 2020: Initial notification

February 19, 2020: Revised submission

March 13, 2020: Final notification

April 12, 2020: Camera ready due

Paper/Thesis Structure by Example

■ Example Paper



[Ahmed Elgohary, Matthias Boehm, Peter J. Haas, Frederick R. Reiss, Berthold Reinwald:
Compressed Linear Algebra for Large-Scale Machine Learning. **PVLDB 2016**]



[Ahmed Elgohary, Matthias Boehm, Peter J. Haas, Frederick R. Reiss, Berthold Reinwald:
Scaling Machine Learning via Compressed Linear Algebra. **SIGMOD Record 2017 46(1)**]



[Ahmed Elgohary, Matthias Boehm, Peter J. Haas, Frederick R. Reiss, Berthold Reinwald:
Compressed Linear Algebra for Large-Scale Machine Learning. **VLDB Journal 2018 27(5)**]

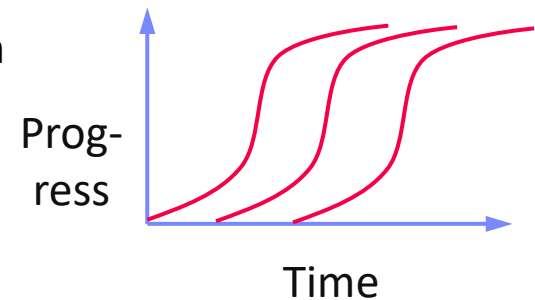


[Ahmed Elgohary, Matthias Boehm, Peter J. Haas, Frederick R. Reiss, Berthold Reinwald:
Compressed Linear Algebra for Large-Scale Machine Learning. **Commun. ACM 2019 62(5)**]

Ideas and Topic Selection

■ Problem-Oriented Research

- Focus on problem/observation first, not your solution
- **Discuss early ideas** with collaborators and friends
- Develop your taste for good research topics
- Topic selection needs time → **pipeline model**



■ Ex. Compressed Linear Algebra

- **Problem:** Iterative ML algorithms + memory-bandwidth-bound operations
→ crucial to fit data in memory → automatic lossless compression
- **Sub-problems:** $\#rows \gg \#cols$, column correlation, column characteristics
→ column-wise compression w/ heterogeneous encoding formats

Prototypes and Experiments

■ Worst Mistake: Schrödinger's Results

- Postpone implementation and experiments till last before the deadline
- No feedback, no reaction time (experiments require many iterations)
- **Karl Popper**: falsifiability of scientific results

■ Continuous Experiments

- Run experiments during survey / prototype building
- Systematic experiments → observations and ideas for improvements
- Don't be afraid of throw away prototypes that don't work

■ Ex. Compressed Linear Algebra

- Data characteristics inspired overall design of encoding schemes
- Initially slow compression → dedicated sampling schemes and estimators
- Initially slow compressed operations → cache-conscious operations, selected operations with better asymptotic behavior

Title and Authors

List of Authors

- #1 by contribution
(main, ..., advisor)
- #2 by last name

Compressed Linear Algebra for Large-Scale Machine Learning

Ahmed Elgohary^{2*}, Matthias Boehm¹, Peter J. Haas¹, Frederick R. Reiss¹,
Berthold Reinwald¹

¹ IBM Research – Almaden; San Jose, CA, USA

² University of Maryland; College Park, MD, USA

Title

- Descriptive yet concise
- Short name if possible
→ easier to cite and discuss

SPOOF: Sum-Product Optimization and Operator Fusion for Large-Scale Machine Learning

Tarek Elgamal^{2*}, Shangyu Luo^{3*}, Matthias Boehm¹, Alexandre V. Evfimievski¹,
Shirish Tatikonda⁴, Berthold Reinwald¹, Prithviraj Sen¹

¹ IBM Research – Almaden; San Jose, CA, USA

² University of Illinois; Urbana-Champaign, IL, USA

³ Rice University; Houston, TX, USA

⁴ Target Corporation; Sunnyvale, CA, USA

MNC: Structure-Exploiting Sparsity Estimation for Matrix Expressions

Johanna Sommer
IBM Germany

Matthias Boehm
Graz University of Technology

Alexandre V. Evfimievski
IBM Research – Almaden

Berthold Reinwald
IBM Research – Almaden

Peter J. Haas
UMass Amherst

Abstract

[Simon Peyton Jones: How to write a great research paper, MSR Cambridge]



ABSTRACT

Large-scale machine learning (ML) algorithms are often iterative, using repeated read-only data access and I/O-bound matrix-vector multiplications to converge to an optimal model. It is crucial for performance to fit the data into single-node or distributed main memory. General-purpose, heavy- and lightweight compression techniques struggle to achieve both good compression ratios and fast decompression speed to enable block-wise uncompressed operations. Hence, we initiate work on compressed linear algebra (CLA), in which lightweight database compression techniques are applied to matrices and then linear algebra operations such as matrix-vector multiplication are executed directly on the compressed representations. We contribute effective column compression schemes, cache-conscious operations, and an efficient sampling-based compression algorithm. Our experiments show that CLA achieves in-memory operations performance close to the uncompressed case and good compression ratios that allow us to fit larger datasets into available memory. We thereby obtain significant end-to-end performance improvements up to 26x or reduced memory requirements.

% 1. State the problem

Large-scale machine learning (ML) algorithms are often iterative, using repeated read-only data access and I/O-bound matrix-vector multiplications to converge to an optimal model. It is crucial for performance to fit the data into single-node or distributed main memory.

% 2. Say why it's an interesting problem

General-purpose, heavy- and lightweight compression techniques struggle to achieve both good compression ratios and fast decompression speed to enable block-wise uncompressed operations.

% 3. Say what your solution achieves

Hence, we initiate work on compressed linear algebra (CLA), in which lightweight database compression techniques are applied to matrices and then linear algebra operations such as matrix-vector multiplication are executed directly on the compressed representations. We contribute effective column compression schemes, cache-conscious operations, and an efficient sampling-based compression algorithm. Our experiments show that CLA achieves in-memory operations performance close to the uncompressed case and good compression ratios that allow us to fit larger datasets into available memory.

% 4. Say what follows from your solution

We thereby obtain significant end-to-end performance improvements up to 26x or reduced memory requirements.

Introduction

- **Context** (1 paragraph)
- **Problems** (1-3 paragraphs)
- **[Existing Work** (1 paragraph)]
- **[Idea** (1 paragraph)]
- **Contributions** (1 paragraph)

Contributions: Our major contribution is to make a case for *compressed linear algebra*, where linear algebra operations are directly executed over compressed matrices. We leverage ideas from database compression techniques and sparse matrix representations. The novelty of our approach is a combination of both, leading towards a generalization of sparse matrix representations and operations. The structure of the paper reflects our detailed technical contributions:

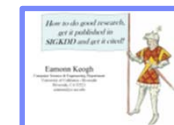
- **Workload Characterization:** We provide the background and motivation for CLA in Section 2 by giving an overview of Apache SystemML, and describing typical linear algebra operations and data characteristics.
- **Compression Schemes:** We adapt several column-based compression schemes to numeric matrices in Section 3 and describe efficient, cache-conscious core linear algebra operations over compressed matrices.
- **Compression Planning:** In Section 4, we further provide an efficient sampling-based algorithm for selecting a good compression plan, including techniques for compressed-size estimation and column grouping.
- **Experiments:** Finally, we integrated CLA into Apache SystemML. In Section 5, we study a variety of full-fledged ML algorithms and real-world datasets in both single-node and distributed settings. We also compare CLA against alternative compression schemes.



Introduction Matters

Anchoring: most reviewers reach their opinion after reading introduction and motivation and then look for evidence

[Eamonn Keogh: How to do good research, get it published in SIGKDD and get it cited!, KDD 2009]



Writing the Paper (and more Experiments)

02 Scientific Reading and Writing
03 Experiments and Reproducibility

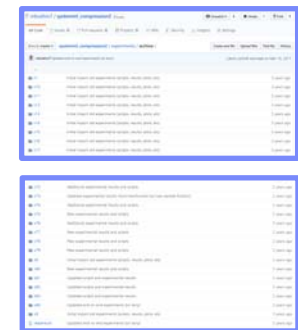
■ Easily Readable: Quality \propto Time

- **Make it easy to skim the paper**
→ paragraph labels, self-explanatory figures (close to text), and structure
- Avoid unnecessary formalism → as simple as possible
- Shortening the text in favor of structure improves readability
- Ex. Compressed Linear Algebra
 - Initial SIGMOD submission: **12+3 pages**
 - Final PVLDB submission: **12 pages**
(+ more figures, experiments, etc)



■ Solid, Reproducible Experiments

- Create, use, and share dedicated benchmarks / datasets
- Avoid weak baselines, start early w/ baseline comparisons
- Automate your experiments as much as possible
- Keep repository of all scripts, results, and used parameters



Related Work

■ Purpose of a “Related Work”-Section

- **Not** a mandatory task or to show you know the field
- Put you work in context of related areas (~ 1 paragraph each)
- Discuss closely related work
- **Crisp separation from existing work** (what are the differences)

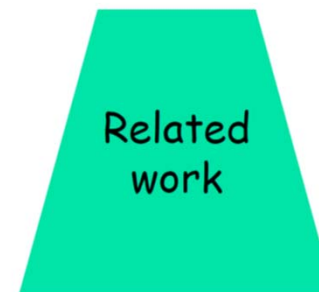
■ Placement

- Section 2 or **Section n-1**
- Throughout the paper

[Simon Peyton Jones: How to write a great research paper, MSR Cambridge]



Your reader



Your idea

■ Give Credit

- Cite broadly, **give credit to inspiring ideas**, create connections
- Honestly acknowledge **limitations of your approach**

- Use LaTeX `\cite{}` and BibTeX
- Use a consistent source of bibtex entries (e.g., DBLP)

```
inproceedings{StonebrakerBPR11,
  author    = {Michael {Stonebraker et al.}},
  title     = {{The Architecture of SciDB}},
  booktitle = {{SSDBM}},
  year      = {2011}
}
```

VLDB2016.bib

```
\bibliographystyle{abbrv}
\bibliography{VLDB2016}
```

- But, **not in footnotes** (unless required)

- [1] M. Abadi et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *CoRR*, 2016.
- [2] A. Alexandrov et al. The Stratosphere Platform for Big Data Analytics. *VLDB J.*, 23(6), 2014.

[AI14] Alexandrov, A. et al.: The Stratosphere platform for big data J. 23/6, 2014.

[AS14] Arap, O.; Swamy, M.: Offloading MPI Parallel Prefix Scan the NetFPGA. CoRR abs/1408.4939/, 2014.

Jaume Amores. 2013. Multiple instance classification: Review, taxonomy and comparative study. *Artificial Intelligence*.

7. CONCLUSIONS

We have initiated work on compressed linear algebra techniques and linear algebra operations are performed directly on the compressed representation. We introduced a new data structure, the compressed sparse matrix, and the compressed matrix-vector multiplication, efficient for compressed matrices, and an efficient sampling-based method for matrix-vector multiplication. The proposed algorithm outperforms code for the uncompressed row and compression matrix-vector multiplication. The proposed algorithm also outperforms other state-of-the-art algorithms for compressed matrix-vector multiplication, such as the high-performance libraries like ScaLapack, providing significant speedup over the uncompressed matrix-vector multiplication. When we have demonstrated the general usefulness of CLA, evidenced by declarative ML that hides the underlying physical processes, such as the CLAs for the sparse matrix-vector multiplication, encoding both dense and sparse matrices in a compressed matrix, and the compressed matrix-vector multiplication, we can extend the CLAs to other applications, such as to any system that provides blockaded matrix representations, such as the quantum circuit simulation. The proposed architecture work includes (1) full optimization, (2) global optimization, (3) hardware architecture, (4) algorithmic compression scheme, and (4) operation level matrix-vector.

8. REFERENCES

- [1] S. Ghemawat, "MapReduce: Large-Scale MapReduce on Heterogeneous Distributed Systems," *CoRR*, 2006.
- [2] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [3] A. A. Chien, "A Survey of Sparse Matrix-Vector Multiplication on GPUs," *Journal of Supercomputing*, vol. 23, no. 1, pp. 1–17, 2012.
- [4] A. A. Aghaj, et al. On Optimizing Machine Learning with Sparse Matrices, *Proceedings of the 17th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2006)*, pp. 109–120, 2006.
- [5] J. Bergstra et al., Theano: A CPU and GPU Math Compiler, *arXiv preprint arXiv:1309.4001*, 2013.
- [6] K. S. Boyer et al., On Synthesizing for Distinct Value Matrices, *Proceedings of the 2014 ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2014)*, pp. 109–120, 2014.
- [7] B. Bhattacharjee et al., Efficient Distinct Value Computation in ORF, *Proceedings of the 2014 ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2014)*, pp. 109–120, 2014.
- [8] S. Bhattacharjee et al., Filter: An Efficient Filtering Framework for Distinct Value Computation, *Proceedings of the 2014 ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2014)*, pp. 109–120, 2014.
- [9] C. Bling et al., Dictionary-based Sparse-Generating String Matrices, *Proceedings of the 2014 ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2014)*, pp. 109–120, 2014.
- [10] S. Ghemawat, "MapReduce: Large-Scale MapReduce on Heterogeneous Distributed Systems," *CoRR*, 2006.
- [11] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [12] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [13] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [14] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [15] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [16] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [17] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [18] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [19] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [20] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [21] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [22] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [23] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [24] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [25] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [26] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [27] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [28] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [29] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [30] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [31] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [32] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [33] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [34] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [35] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [36] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [37] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [38] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [39] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [40] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [41] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [42] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [43] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [44] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [45] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [46] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [47] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [48] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [49] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [50] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [51] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [52] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [53] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [54] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [55] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [56] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [57] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [58] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [59] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [60] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [61] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [62] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [63] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [64] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [65] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [66] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [67] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [68] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [69] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [70] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [71] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [72] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [73] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [74] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [75] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [76] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [77] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [78] S. Ghemawat, "MapReduce: Simplified Data Processing on Big Data Analysis," *VLDB J.*, 2010, 2014.
- [79] S. Ghemawat, "Map

Dealing with Feedback / Criticism

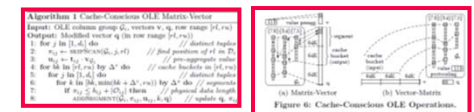
Different Kinds of Feedback

- Casual discussion of early ideas
- Comments on paper drafts
- Reviewer comments (good and bad)

Always welcome
feedback/criticism
Address all feedback w/
sincere effort

Example Compressed Linear Algebra

- SIGMOD Reviewer 2 (REJECT)
 - “The rewriting for $q=Xv$ seems wrong: To compute q , one takes each row of the matrix X and multiplies it with the vector v .”
- PVLDB Reviewer 3 (WEAK ACCEPT)
 - “I kinda disagree with the broad definition of declarative ML from the introduction.”



Algorithm 1 Cache-Conscious OLE: Matrix-Vector
Input: OLE column group G , vector v , q , row range $[r, r+1]$
Output: Modified vector q (in row range $[r, r+1]$)
1: for j in $[1, r]$ do
2: $q_j = v_j$ // distinct input
3: $q_j = v_j + \Delta v_j$ // pre-aggregate value
4: for j in $[r+1, r+1]$ do
5: for k in G such that $\Delta v_k \neq 0$ do // distinct inputs
6: $q_j = v_j + \Delta v_k$ // pre-aggregate value
7: $q_j = v_j + \Delta v_k$ // pre-aggregate value
8: $q_j = v_j + \Delta v_k$ // pre-aggregate value
9: $q_j = v_j + \Delta v_k$ // pre-aggregate value
10: $q_j = v_j + \Delta v_k$ // pre-aggregate value
11: $q_j = v_j + \Delta v_k$ // pre-aggregate value
12: $q_j = v_j + \Delta v_k$ // pre-aggregate value
13: $q_j = v_j + \Delta v_k$ // pre-aggregate value
14: $q_j = v_j + \Delta v_k$ // pre-aggregate value
15: $q_j = v_j + \Delta v_k$ // pre-aggregate value
16: $q_j = v_j + \Delta v_k$ // pre-aggregate value
17: $q_j = v_j + \Delta v_k$ // pre-aggregate value
18: $q_j = v_j + \Delta v_k$ // pre-aggregate value
19: $q_j = v_j + \Delta v_k$ // pre-aggregate value
20: $q_j = v_j + \Delta v_k$ // pre-aggregate value
21: $q_j = v_j + \Delta v_k$ // pre-aggregate value
22: $q_j = v_j + \Delta v_k$ // pre-aggregate value
23: $q_j = v_j + \Delta v_k$ // pre-aggregate value
24: $q_j = v_j + \Delta v_k$ // pre-aggregate value
25: $q_j = v_j + \Delta v_k$ // pre-aggregate value
26: $q_j = v_j + \Delta v_k$ // pre-aggregate value
27: $q_j = v_j + \Delta v_k$ // pre-aggregate value
28: $q_j = v_j + \Delta v_k$ // pre-aggregate value
29: $q_j = v_j + \Delta v_k$ // pre-aggregate value
30: $q_j = v_j + \Delta v_k$ // pre-aggregate value
31: $q_j = v_j + \Delta v_k$ // pre-aggregate value
32: $q_j = v_j + \Delta v_k$ // pre-aggregate value
33: $q_j = v_j + \Delta v_k$ // pre-aggregate value
34: $q_j = v_j + \Delta v_k$ // pre-aggregate value
35: $q_j = v_j + \Delta v_k$ // pre-aggregate value
36: $q_j = v_j + \Delta v_k$ // pre-aggregate value
37: $q_j = v_j + \Delta v_k$ // pre-aggregate value
38: $q_j = v_j + \Delta v_k$ // pre-aggregate value
39: $q_j = v_j + \Delta v_k$ // pre-aggregate value
40: $q_j = v_j + \Delta v_k$ // pre-aggregate value
41: $q_j = v_j + \Delta v_k$ // pre-aggregate value
42: $q_j = v_j + \Delta v_k$ // pre-aggregate value
43: $q_j = v_j + \Delta v_k$ // pre-aggregate value
44: $q_j = v_j + \Delta v_k$ // pre-aggregate value
45: $q_j = v_j + \Delta v_k$ // pre-aggregate value
46: $q_j = v_j + \Delta v_k$ // pre-aggregate value
47: $q_j = v_j + \Delta v_k$ // pre-aggregate value
48: $q_j = v_j + \Delta v_k$ // pre-aggregate value
49: $q_j = v_j + \Delta v_k$ // pre-aggregate value
50: $q_j = v_j + \Delta v_k$ // pre-aggregate value
51: $q_j = v_j + \Delta v_k$ // pre-aggregate value
52: $q_j = v_j + \Delta v_k$ // pre-aggregate value
53: $q_j = v_j + \Delta v_k$ // pre-aggregate value
54: $q_j = v_j + \Delta v_k$ // pre-aggregate value
55: $q_j = v_j + \Delta v_k$ // pre-aggregate value
56: $q_j = v_j + \Delta v_k$ // pre-aggregate value
57: $q_j = v_j + \Delta v_k$ // pre-aggregate value
58: $q_j = v_j + \Delta v_k$ // pre-aggregate value
59: $q_j = v_j + \Delta v_k$ // pre-aggregate value
60: $q_j = v_j + \Delta v_k$ // pre-aggregate value
61: $q_j = v_j + \Delta v_k$ // pre-aggregate value
62: $q_j = v_j + \Delta v_k$ // pre-aggregate value
63: $q_j = v_j + \Delta v_k$ // pre-aggregate value
64: $q_j = v_j + \Delta v_k$ // pre-aggregate value
65: $q_j = v_j + \Delta v_k$ // pre-aggregate value
66: $q_j = v_j + \Delta v_k$ // pre-aggregate value
67: $q_j = v_j + \Delta v_k$ // pre-aggregate value
68: $q_j = v_j + \Delta v_k$ // pre-aggregate value
69: $q_j = v_j + \Delta v_k$ // pre-aggregate value
70: $q_j = v_j + \Delta v_k$ // pre-aggregate value
71: $q_j = v_j + \Delta v_k$ // pre-aggregate value
72: $q_j = v_j + \Delta v_k$ // pre-aggregate value
73: $q_j = v_j + \Delta v_k$ // pre-aggregate value
74: $q_j = v_j + \Delta v_k$ // pre-aggregate value
75: $q_j = v_j + \Delta v_k$ // pre-aggregate value
76: $q_j = v_j + \Delta v_k$ // pre-aggregate value
77: $q_j = v_j + \Delta v_k$ // pre-aggregate value
78: $q_j = v_j + \Delta v_k$ // pre-aggregate value
79: $q_j = v_j + \Delta v_k$ // pre-aggregate value
80: $q_j = v_j + \Delta v_k$ // pre-aggregate value
81: $q_j = v_j + \Delta v_k$ // pre-aggregate value
82: $q_j = v_j + \Delta v_k$ // pre-aggregate value
83: $q_j = v_j + \Delta v_k$ // pre-aggregate value
84: $q_j = v_j + \Delta v_k$ // pre-aggregate value
85: $q_j = v_j + \Delta v_k$ // pre-aggregate value
86: $q_j = v_j + \Delta v_k$ // pre-aggregate value
87: $q_j = v_j + \Delta v_k$ // pre-aggregate value
88: $q_j = v_j + \Delta v_k$ // pre-aggregate value
89: $q_j = v_j + \Delta v_k$ // pre-aggregate value
90: $q_j = v_j + \Delta v_k$ // pre-aggregate value
91: $q_j = v_j + \Delta v_k$ // pre-aggregate value
92: $q_j = v_j + \Delta v_k$ // pre-aggregate value
93: $q_j = v_j + \Delta v_k$ // pre-aggregate value
94: $q_j = v_j + \Delta v_k$ // pre-aggregate value
95: $q_j = v_j + \Delta v_k$ // pre-aggregate value
96: $q_j = v_j + \Delta v_k$ // pre-aggregate value
97: $q_j = v_j + \Delta v_k$ // pre-aggregate value
98: $q_j = v_j + \Delta v_k$ // pre-aggregate value
99: $q_j = v_j + \Delta v_k$ // pre-aggregate value
100: $q_j = v_j + \Delta v_k$ // pre-aggregate value

[Matthias Boehm et al.:
Declarative Machine
Learning - A Classification
of Basic Properties and
Types. **CoRR 2016.**]



Paper Rebuttal and/or Revision

- Rebuttal**: seriously consider all feedback (in doubt agree), and answer with facts / ideas how to address the comments
- Revision** (conditional accept): address all revision requests

Summary and Q&A

- Data Management Group
- Course Organization, Outline, and Projects
- Structure of Scientific Papers

- Remaining Questions?

- Next Lectures
 - 02 Scientific Reading and Writing [Nov 05]
 - 03 Experiments, Reproducibility, and Projects [Nov 12]