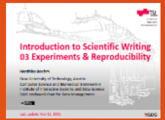
Seminar Large-scale Data Engineering (LDE)
03 Experiments, Reproducibility & Presentations

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

Last update: May 8, 2023

[**Credit:** Based on “Introduction to Scientific Writing”/ ”03 Experiments and Reproducibility” by Matthias Boehm (TU Graz, winter 2021/22)]



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

- **Finished Selection of Seminar Topics/Papers**
  - All participants have selected their seminar topic

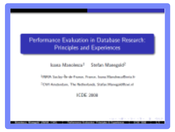
- **Exam Registration via MTS**
  - For seminar-only, project-only, seminar+project modules LDE
  - Currently being set up, not open yet
  - Details will be announced on the ISIS page in the next days

Agenda

- **Experiments and Result Presentation**
- **Reproducibility and RDM**
- **Scientific Presentations**

Experiments and Result Presentation

In Computer Science (Data Management)



[**Credit:** Ioana Manolescu, Stefan Manegold: Performance Evaluation in Database Research: Principles and Experiences, **ICDE 2008**]

Motivation

- **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 throwing away prototypes that don’t work
- **Good Research Fires Itself**
  - Initial experiments give directions for further improvements
  - Problem-oriented methodology

Types of Experiments

- **#1 Exploratory Experiments**
  - Tests for functional correctness
  - Unstructured experiments for initial feedback → eval feasibility
- **#2 Micro Benchmarks**
  - Measure specific aspects in controlled and understandable scope
  - Bottom-up approach
- **#3 Benchmarks**
  - Evaluate on community/own benchmarks
  - Examples: TPC-C, TPC-H, TPC-DS, JOB, MLPerf
- **#4 End-to-end Applications**
  - Evaluate in larger scope of real datasets and query workloads
  - Examples: Customer workload, ML pipelines (dataprep, training, eval)

From Idea to Experiments

[I. Manolescu, S/ Manegold: Performance Evaluation in Database Research: Principles and Experiences, **ICDE 2008**]

- **Overview**
  - Proper planning helps to keep you from "getting lost"
  - Repeatable experiments simplify your own work
  - There is **no single way** how to **do it right**
  - There are **many ways** how to **do it wrong**

- **Basic Planning**
  - Which data / data sets should be used?
  - Which workload / queries should be run?
  - Which hardware & software should be used?
  - **Metrics:** What to measure? How to measure?
  - **Comparison:** How to compare? CSI: How to find out what is going on?

Dataset Selection

- **Synthetic Data**
  - Generate data with specific data characteristics
  - Systematic evaluation w/ data size, sparsity, etc.

Representative of real data distributions?

- **“Real” Data Repositories**
  - Wide selection of available datasets w/ different characteristics
  - UCI ML Repository: https://archive.ics.uci.edu/ml/index.php
  - Florida Sparse Matrix Collection: https://sparse.tamu.edu/
  - Google dataset search: https://datasetsearch.research.google.com/
  - Common Datasets in ML: ImageNet, Mnist, CIFAR, KDD, Criteo
  - Common Datasets in DM: Census, Taxi, Airlines, DBLP, benchmarks etc.

Representative for variety of workloads / common case?

Benchmarks

- **Overview**
  - Community- and organization-driven creation of agreed benchmarks
  - **Benchmarks can define a field** and foster innovation
- **#1 Data Management**
  - Query processing: 007, TPC-C, TPC-E, TPC-H, TPC-DS (w/ audit)
  - Join ordering: JOB
- **#2 “Big Data”**
  - MR/Spark: BigBench, HiBench, SparkBench
  - Array Databases: GenBase
- **#3 Machine Learning Systems**
  - SLAB, DAWNBench, MLPerf, MLBench, AutoML Bench, Meta Worlds, TPCx-AI

[Michael J. Carey, David J. DeWitt, Jeffrey F. Naughton: The oo7 Benchmark. **SIGMOD 1993**]

[http://www.tpc.org/tpch/]

(See AMLS course for details)

Benchmarks, cont.

[**MLPerf** v0.6: https://mlperf.org/training-results-0-6/]

Closed Division Times

#	Submitter	System	Processor	#	Accelerator	#	Software	Benchmark results (minutes)							Details	Code	Notes
								Image classification	Object detection, light-weight	Object detection, heavy-wt.	Translation, recurrent	Translation, non-recur.	Recommendation	Reinforcement Learning			
								ImageNet	COCO	COCO	WMT E-G	WMT E-G	MovieLens-20M	Go			
								ResNet-50 v1.5	SSD w/ ResNet-34	Mask-R-CNN	NMT	Transformer	NCF	Mini Go			
Available in cloud																	
0.6-1	Google	TPUv3.32			TPUv3	16	TensorFlow, TPU 1.14.1.dev	42.19	12.61	107.03	12.25	10.20	[1]		details	code	none
0.6-2	Google	TPUv3.128			TPUv3	64	TensorFlow, TPU 1.14.1.dev	11.22	3.89	57.46	4.62	3.85	[1]		details	code	none
0.6-3	Google	TPUv3.256			TPUv3	128	TensorFlow, TPU 1.14.1.dev	6.86	2.76	35.60	3.53	2.81	[1]		details	code	none
0.6-4	Google	TPUv3.512			TPUv3	256	TensorFlow, TPU 1.14.1.dev	3.85	1.79		2.51	1.58	[1]		details	code	none
0.6-5	Google	TPUv3.1024			TPUv3	512	TensorFlow, TPU 1.14.1.dev	2.27	1.34		2.11	1.05	[1]		details	code	none
0.6-6	Google	TPUv3.2048			TPUv3	1024	TensorFlow, TPU 1.14.1.dev	1.28	1.21			0.85	[1]		details	code	none
Available on-premise																	
0.6-7	Intel	32x 2S CLX 8260L	CLX 8260L	64			TensorFlow						[1]	14.43	details	code	none
0.6-8	NVIDIA	DGX-1			Tesla V100	8	MXNet, NGC19.05	115.22					[1]		details	code	none
0.6-9	NVIDIA	DGX-1			Tesla V100	8	PyTorch, NGC19.05		22.36	207.48	20.55	20.34	[1]		details	code	none
0.6-10	NVIDIA	DGX-1			Tesla V100	8	TensorFlow, NGC19.05						[1]	27.39	details	code	none
0.6-11	NVIDIA	3x DGX-1			Tesla V100	24	TensorFlow, NGC19.05						[1]	13.57	details	code	none
0.6-12	NVIDIA	24x DGX-1			Tesla V100	192	PyTorch, NGC19.05			22.03			[1]		details	code	none
0.6-13	NVIDIA	30x DGX-1			Tesla V100	240	PyTorch, NGC19.05		2.67				[1]		details	code	none
0.6-14	NVIDIA	48x DGX-1			Tesla V100	384	PyTorch, NGC19.05				1.99		[1]		details	code	none
0.6-15	NVIDIA	60x DGX-1			Tesla V100	480	PyTorch, NGC19.05					2.05	[1]		details	code	none
0.6-16	NVIDIA	130x DGX-1			Tesla V100	1040	MXNet, NGC19.05	1.69					[1]		details	code	none
0.6-17	NVIDIA	DGX-2			Tesla V100	16	MXNet, NGC19.05	57.87					[1]		details	code	none
0.6-18	NVIDIA	DGX-2			Tesla V100	16	PyTorch, NGC19.05		12.21	101.00	10.94	11.04	[1]		details	code	none
0.6-19	NVIDIA	DGX-2H			Tesla V100	16	MXNet, NGC19.05	52.74					[1]		details	code	none
0.6-20	NVIDIA	DGX-2H			Tesla V100	16	PyTorch, NGC19.05		11.41	95.20	9.87	9.80	[1]		details	code	none
0.6-21	NVIDIA	4x DGX-2H			Tesla V100	64	PyTorch, NGC19.05		4.78	32.72			[1]		details	code	none
0.6-22	NVIDIA	10x DGX-2H			Tesla V100	160	PyTorch, NGC19.05					2.41	[1]		details	code	none
0.6-23	NVIDIA	12x DGX-2H			Tesla V100	192	PyTorch, NGC19.05			18.47			[1]		details	code	none
0.6-24	NVIDIA	15x DGX-2H			Tesla V100	240	PyTorch, NGC19.05		2.56				[1]		details	code	none
0.6-25	NVIDIA	16x DGX-2H			Tesla V100	256	PyTorch, NGC19.05				2.12		[1]		details	code	none
0.6-26	NVIDIA	24x DGX-2H			Tesla V100	384	PyTorch, NGC19.05				1.80		[1]		details	code	none
0.6-27	NVIDIA	30x DGX-2H, 8 chips each			Tesla V100	240	PyTorch, NGC19.05		2.23				[1]		details	code	none
0.6-28	NVIDIA	30x DGX-2H			Tesla V100	480	PyTorch, NGC19.05					1.59	[1]		details	code	none
0.6-29	NVIDIA	32x DGX-2H			Tesla V100	512	MXNet, NGC19.05	2.59					[1]		details	code	none
0.6-30	NVIDIA	96x DGX-2H			Tesla V100	1536	MXNet, NGC19.05	1.33					[1]		details	code	none

96 x DGX-2H

= 96 * 16 = 1536 V100 GPUs

➔ ~ 96 * $400K = **$35M – $40M**

[https://www.forbes.com/sites/tiriasresearch/2019/06/19/nvidia-offers-a-turnkey-supercomputer-the-dgx-superpod/#693400f43ee5]

Baselines

- **#1 Primary Baseline**
  - Existing algorithm or system infrastructure
  - Main comparison point, usually with same runtime operations
  - **Beware:** Avoid speedup-only results (need absolute numbers for grounding)
- **#2 Additional Baselines**
  - Alternative systems with different runtime and compiler
  - Usually, not directly comparable but important for grounding
  - E.g., SystemDS: R, Julia, Spark, TensorFlow, PyTorch
  - **Potential hindering factors:** commercial and closed-source systems, software licenses

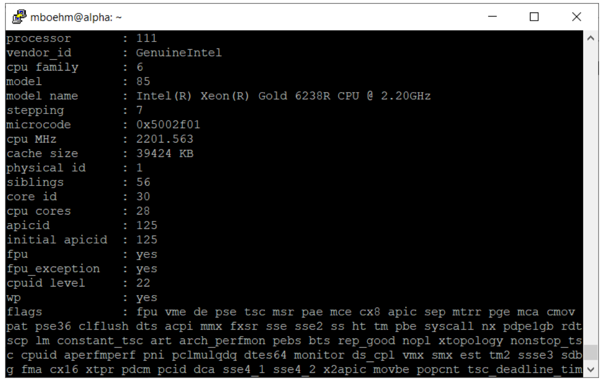
- **Problem of Weak Baselines**
  - Authors want to show improvements
  - Successive improvements over state-of-the-art don't add up

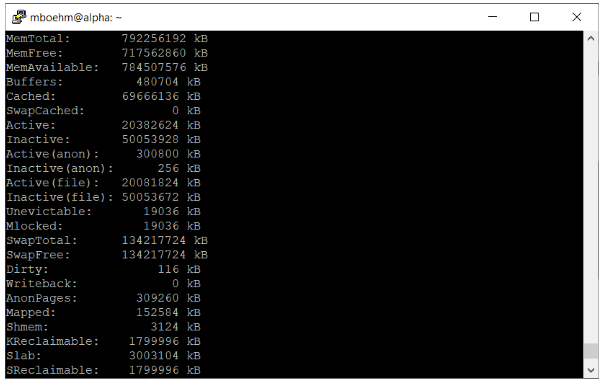
[Timothy G. Armstrong, Alistair Moffat, William Webber**,** Justin Zobel: Improvements That Don't Add Up: Ad-Hoc Retrieval Results Since 1998. **CIKM 2009**]

[Maurizio Ferrari Dacrema, Paolo Cremonesi, Dietmar Jannach: Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. **RecSys 2019**]

Presentation – Experimental Setting

- **Hardware Selection**
  - Multiple nodes for distributed computation
  - Avoid too outdated HW (irrelevance)
- **Find Balanced Level of Detail**
  - Underspecified:
    `“We ran all experiments on an Intel CPU”`
  - Over-specified:
    `cat /proc/cpuinfo`
    `cat /proc/meminfo`

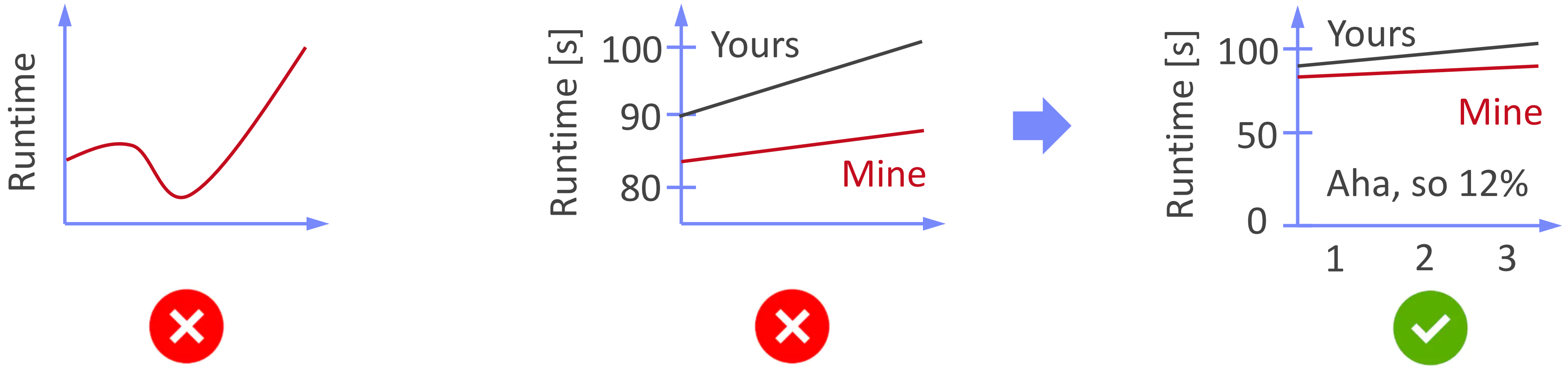




- **Recommendation**
  - **HW components:** #nodes, CPUs (#cores, clock freq., cache size), memory, network, I/O
  - **SW components:** OS, programming language, versions, other software, compiler flags
  - Baselines and configuration → Use **recent versions of baseline systems**
  - Data and workloads with data sizes, parameters, configurations

Presentation – Figures

- **Axes**
  - Use informative axes labels with units (e.g., Total Execution Time [ms])
  - Don’t cheat or mislead readers and reviewers
  - Start y-axis at 0 for linear scale

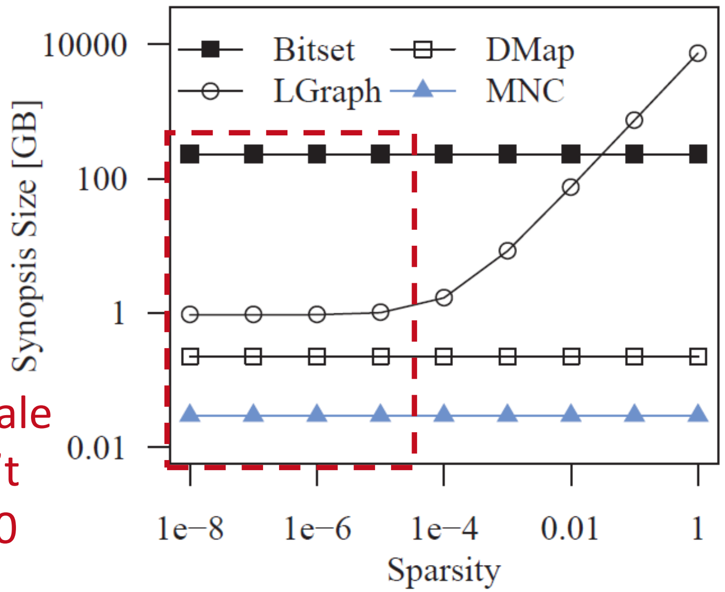


What are the units?
Where are the tics?

Misleading y axis

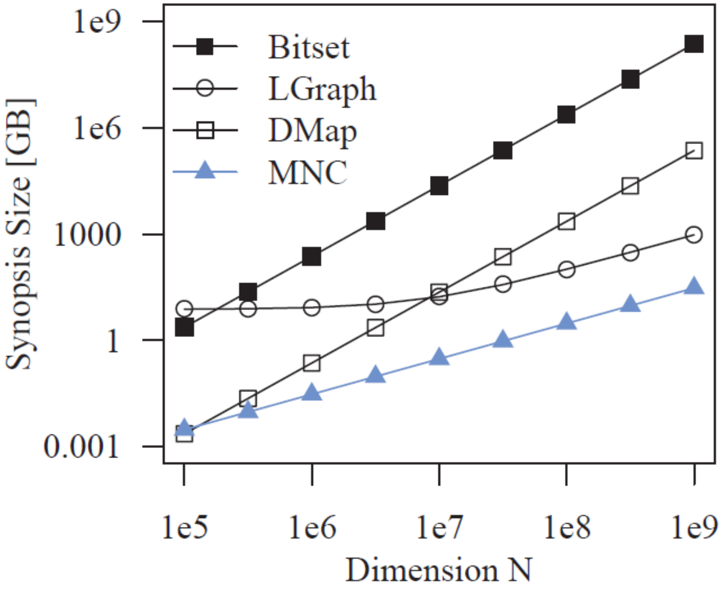
Presentation – Figures, cont.

- **Fair Ranges of Parameters**
  - Evaluate common ranges of values
  - Don't hide important information



For log-scale you can't start at 0

Don't limit range to make you look good



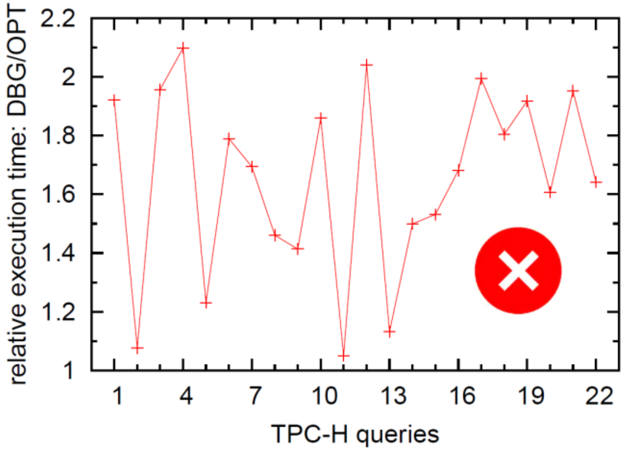
[J. Sommer, M. Boehm, A. V. Evfimievski, B. Reinwald, P. J. Haas: MNC: Structure-Exploiting Sparsity Estimation for Matrix Expressions. **SIGMOD 2019**]

If there are multiple relevant parameters, show them all

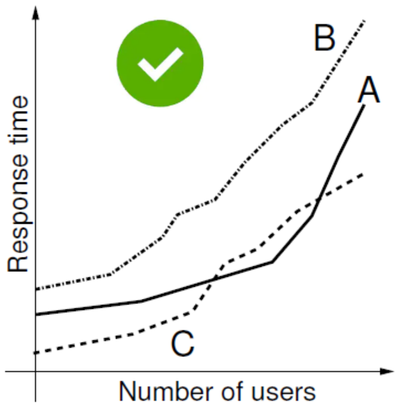
Presentation – Figures, cont.

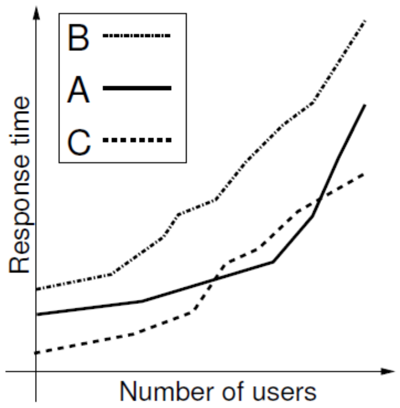
[I. Manolescu, S/ Manegold: Performance Evaluation in Database Research: Principles and Experiences, **ICDE 2008**]

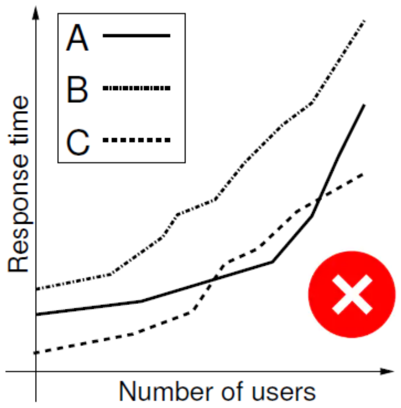
- **Plots Types**
  - **Barplot** for categories
  - **Plot** + Line/linepoints for continuous parameters
  - Visible font sizes (similar to text)



- **Legends**
  - Order them by appearance
  - Attach directly to graph



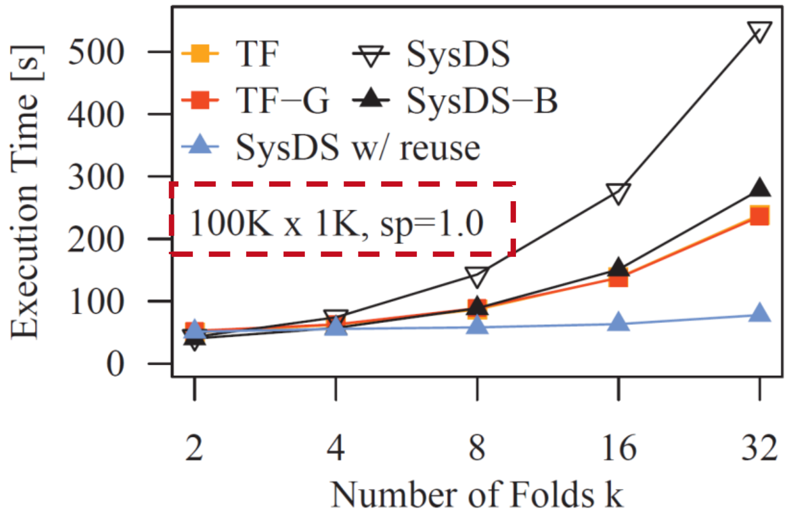


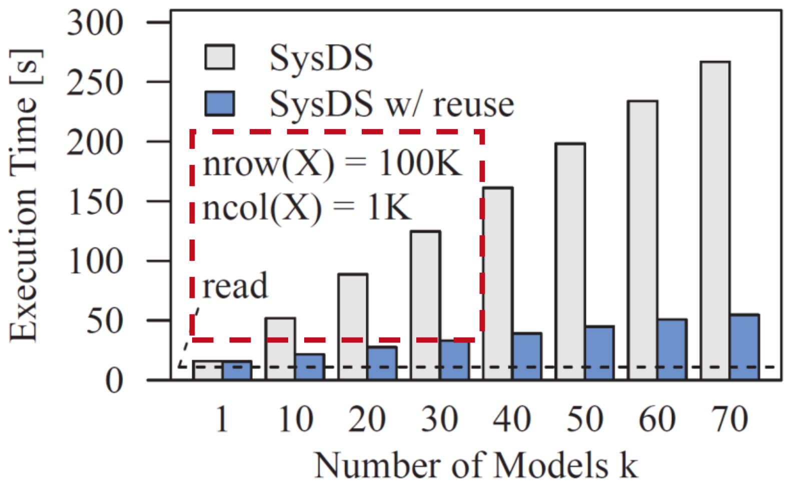


Human brain is a poor join processor
Humans get frustrated

Presentation – Figures, cont.

- **Diversity & Consistency**
  - Diversity: if applicable use mix of different plot types and tables
  - Consistency: use consistent colors and names for same baselines
- **Labeling**
  - Make the plots self-contained
  - Simplifies skimming and avoids join with text

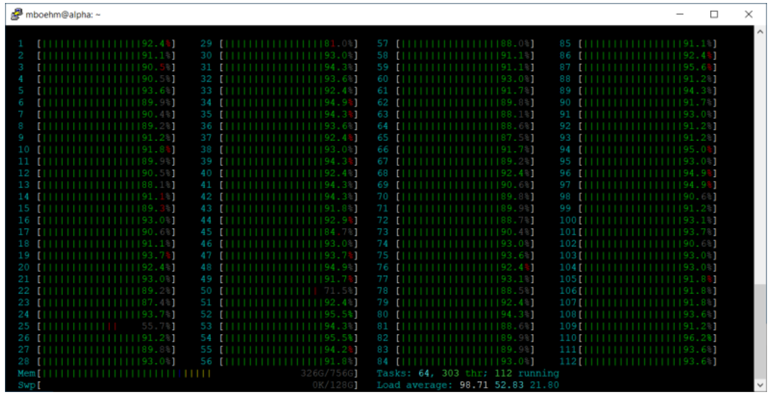




[Matthias Boehm et al: SystemDS: A Declarative Machine Learning System for the End-to-End Data Science Lifecycle. **CIDR 2020**]

Presentation – Result Interpretation

- **Use the Right OS Tools**
  - System-specific tracing/statistics
  - `top / htop / iotop`
    (looks **CPU bound**)
  - `perf -stat -d ./run.sh`
    (no, it's **memory-bandwidth bound**)

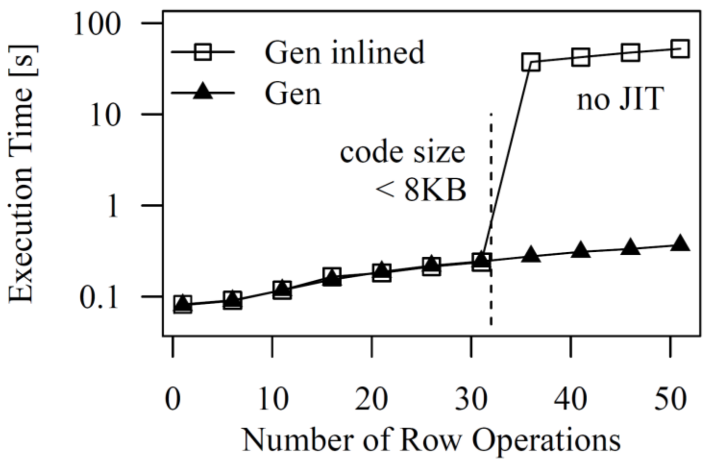


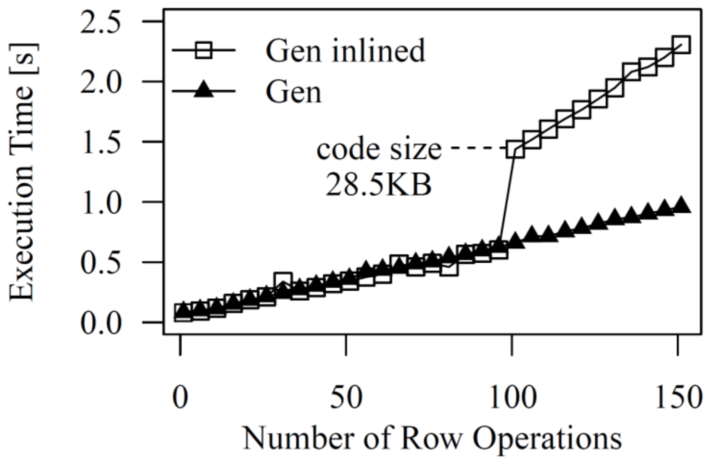
```
Performance counter stats for './run.sh':
     12721364.53 msec task-clock            #   83.640 CPUs utilized
          463352      context-switches      #    0.036 K/sec
   5455536095415      instructions          #    0.14  insn per cycle           (62.50%)
    335314473273      branches              #   26.358 M/sec
          (62.50%)
      1463380955      branch-misses         #    0.44% of all branches          (62.50%)
   2185062643097      L1-dcache-loads       #  171.763 M/sec                    (62.50%)
    142845949268      L1-dcache-load-misses #    6.54% of all L1-dcache hits    (62.50%)
      3375555316      LLC-loads             #    0.265 M/sec                    (50.00%)
      1016330404      LLC-load-misses       #   30.11% of all LL-cache hits     (50.00%)

   152.096000108 seconds time elapsed
 12052.466691000 seconds user
   674.704421000 seconds sys
```

Don't just report the results, try to understand and explain them

Presentation – Result Interpretation, cont.

- **Use the Right PL Tools / Flags**
  - E.g., Understanding Java JIT compilation `-XX:+PrintCompilation`
  - E.g., Understanding HW Cache Hierarchy (L1i 32KB) `-XX:-DontCompileHugeMethods`





[Matthias Boehm et al: On Optimizing Operator Fusion Plans for Large-Scale Machine Learning in SystemML. **PVLDB 11(12) 2018**]

Reproducibility and RDM (Research Data Management)

In Computer Science (Data Management)

Research Data Management (RDM)

- **Overview**
  - Ensure reproducibility of research results and conclusions

  - **Common problem:** `“All code and data was on the student’s laptop and the student left / the laptop crashed.”`

  - **Create value for others** (compare, reuse, understand, extend)
  - EU Projects: Mandatory proposal section & deliverable on RDM plan

Excursus: FAIR Data Principles

[https://www.go-fair.org/fair-principles/]

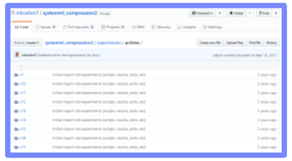
- **#1 Findable**
  - Metadata and data have globally unique **persistent identifiers**
  - Data described with rich **meta data**; registered/indexes and searchable
- **#2 Accessible**
  - Metadata and data retrievable via open, free and universal **comm protocols**
  - Metadata accessible even when data no longer available
- **#3 Interoperable**
  - Metadata and data use a formal, **accessible, and broadly applicable format**
  - Metadata and data use FAIR vocabularies and qualified references
- **#4 Reusable**
  - Metadata and data described with plurality of accurate and relevant attributes
  - Clear license, **associated with provenance**, meets community standards

RDM in Practice @ DAMS Lab

- **Code and Artifacts**
  - Open-source systems
    - Apache SystemDS: https://github.com/apache/systemds
    - DAPHNE: https://github.com/daphne-eu/daphne
    - Complete code history, src/bin releases
    - LDE/AMLS/DIA programming projects in SystemDS and DAPHNE
  - Additional private GitHub repos for student projects / prototypes
  - Reproducibility for publications: https://github.com/damslab/reproducibility

- **Central Paper Repository**
  - All paper submissions with LaTeX sources, figures, reviews, rebuttals, etc.
  - All paper-related experiments
    - `Archive:` append-only experimental results
    - `Plots:` scripts and figures of plots
    - `Results:` latest results used for the current plots
    - `Scripts:` data preparation, baselines, benchmarks

➔ Automate your experiments as much as possible



SIGMOD Reproducibility Process

- **Overview**
  - Accepted papers can submit package, verified by committee
  - ACM Results Replicated / ACM Artifacts Available labels
  - Most Reproducible Paper Award ($750, visibility)

- **#1 Replicability (aka Repeatability)**
  - Recreate result data and graphs shown in the final paper
  - **Expected:** same trend of baseline comparisons, parameter influence

- **#2 Reproducibility**
  - Verify robustness of results wrt parameters and environments
  - **Examples:** different data and workload characteristics, hardware

SIGMOD Reproducibility Process, cont.

- **Ideal Reproducibility Submission**

  "At a minimum the authors should provide a complete set of scripts to install the system, produce the data, run experiments and produce the resulting graphs along with a detailed Readme file that describes the process step by step so it can be easily reproduced by a reviewer.

  The ideal reproducibility submission consists of a master script that:

  1. installs all systems needed,
  2. generates or fetches all needed input data,
  3. reruns all experiments and generates all results,
  4. generates all graphs and plots, and finally,
  5. recompiles the sources of the paper

  ... to produce a new PDF for the paper that contains the new graphs. "

[**Credit:** db-reproducibility.seas.harvard.edu/#Guidelines]

- **Note: It takes time, plan from start**
  - We prepared for SIGMOD 2019 Repro, but finally, not submitted (ran out of time)

[J. Sommer, M. Boehm, A. V. Evfimievski, B. Reinwald, P. J. Haas: MNC: Structure-Exploiting Sparsity Estimation for Matrix Expressions. **SIGMOD 2019**]

Tools for Automation

- **Motivation**
  - Tooling for running artifacts in a self-contained manner
  - Metadata storage in semi-structured formats like JSON/XML
- **Examples**

Name	Target	Link
CK	ML Systems	http://ctuning.org
CWL	Analysis Workflows	http://commonwl.org
Popper	Container Workflows	https://github.com/systemslab/popper
ReproZip	General-purpose Bundles	http://reprozip.org
Sciunit	Self-contained Experiments Bundles	http://sciunit.run
Sumatra	Numerical Simulations	https://github.com/open-research/sumatra

Scientific Presentations

In Computer Science (Data Management)

Goals and Structure

- **Typical Goals of a Scientific Presentation**
  - Make audience aware of and interested in your work → visibility, get your paper cited
  - Show that you made significant contributions to relevant research area
  - Discuss problem/topic, get feedback from audience, foundation for offline discussion

- **Structure**
  - No single best structure, but best practices
  - Commonalities with scientific papers
    - Introduction/motivation (including necessary background and related work)
    - Main part (your own contributions)
    - Experimental results
    - Summary & outlook

Limit the Scope

- **Typical Duration**
  - Lecture: ~90-120 min
  - Thesis defense: ~45 min
  - **Seminar talk: ~20 min**
  - **Conference talk ~5-15 min**

→ **Challenge: Usually not to fill the time, but to not go over time**

- **Limit the Scope, You Cannot Talk about Everything**
  - In terms of breadth
    - E.g., focus on a subset of the contributions
    - E.g., not all experiments
    - But: give overview of everything
  - In terms of depth
    - I.e., don't show all details
    - Present simplified results

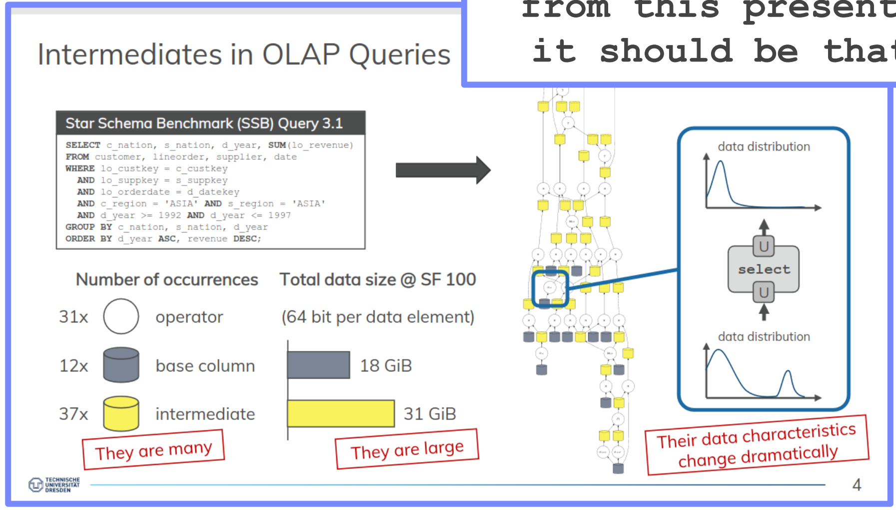
→ **Challenge: Select the most important aspects**

Know Your Audience

- **Audience Characteristics**
  - How much can you expect them to know about your topic? Don’t assume too much…
  - Need to adjust to you as a speaker in the beginning
- **Help Audience Not to Get Lost**
  - Clear motivation (don’t rush through it)
  - Clear presentation outline (after motivation, otherwise hard to comprehend)
  - Outline and current position again after each section of the talk
  - Repeat important assumptions
  - Illustrate theory with concrete (running) examples
  - Take necessary time for complex diagrams, formulas, etc.

- **Steer Audience’s Attention**
  - Make the most important points pop out

“If you remember one thing from this presentation, it should be that …”



  - (Simple) animations
    - Reveal complex slide contents incrementally
    - But: avoid “Powerpoint Poisoning”

Slide Design

- **Avoid Slides Full of Text**
- **Avoid Complex Formulas and Source Code**
  - Unless they really contribute to the understanding
- **Avoid too Small Font Size**
- **Avoid too Many Effects**
- **Use Varied Layouts**
  - Not just lists of bullet points
  - Convey information in diagrams, figures, etc.
- **Use Simple Set of Colors**
- **Use Conscious Line Breaks**

- **Don't Simply Reuse the Figures from Your Paper**

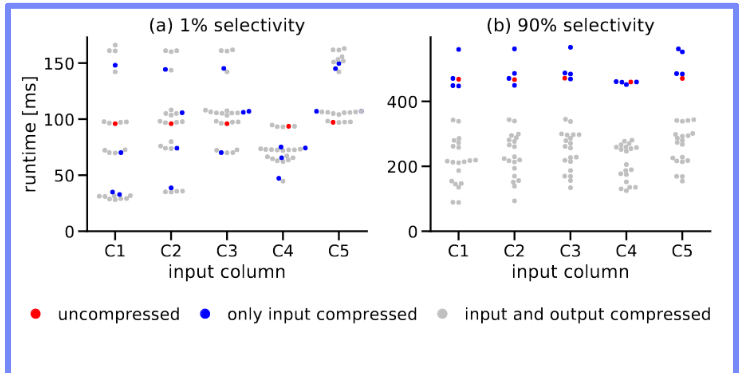
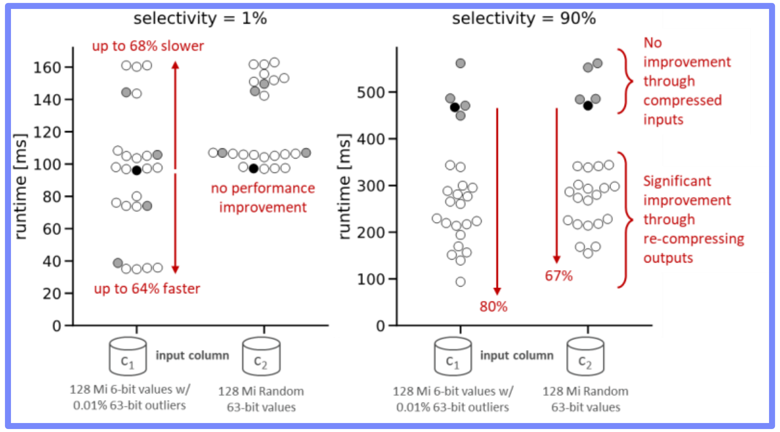


Figure 5: select with on-the-fly de/re-compression.

paper

slides



Preparing a Presentation

- **#1 Planning**
  - What are the key takeaways you want to convey?
  - Structure of the talk, running examples
- **#2 Slide Creation**
  - Create initial slide deck (doesn’t need to be pixel perfect yet)
  - Should contain all planned content, consciously divided into slides
- **#3 Practice**
  - Rehearse aloud, ideally with audience (ask for maximum criticism)
  - Does the timing fit? Is the talk comprehensible?

Iterate steps
if necessary

- **#4 Slide Finalization**
  - Make the slides pixel-perfect

Handling Questions

- **Basic Mindset**
  - **Always welcome questions as well as feedback/criticism**
  - **Take all questions constructively**
- **You Don’t Need to Always Have an Answer**
  - Answer as good as you can
  - Honestly admit if you don’t know the answer, e.g., if it needs further investigation
- **Take longer discussions offline**
  - Don’t bore the rest of the audience with too specific discussions
- **Page Numbers on Slides**
  - Help audience to refer to specific point in your presentation
- **Prepare Back-up Slides**
  - Extra slides not shown in the main presentation
  - Can be useful when answering questions

Summary and Q&A

- **Experiments and Result Presentation**
- **Reproducibility and RDM**
- **Scientific Presentations**

- **Remaining Questions?**
- **Summary Paper Submission: Deadline: Jun 19, 2023, 23:59 CEST**

- **Optional Office Hours**
  - Mondays 14:00 – 18:00 in TEL 811 (and zoom) → ask any seminar-related questions
- **Seminar Presentations**
  - Starting **June 26,** 2023, **in-person only (no zoom)**