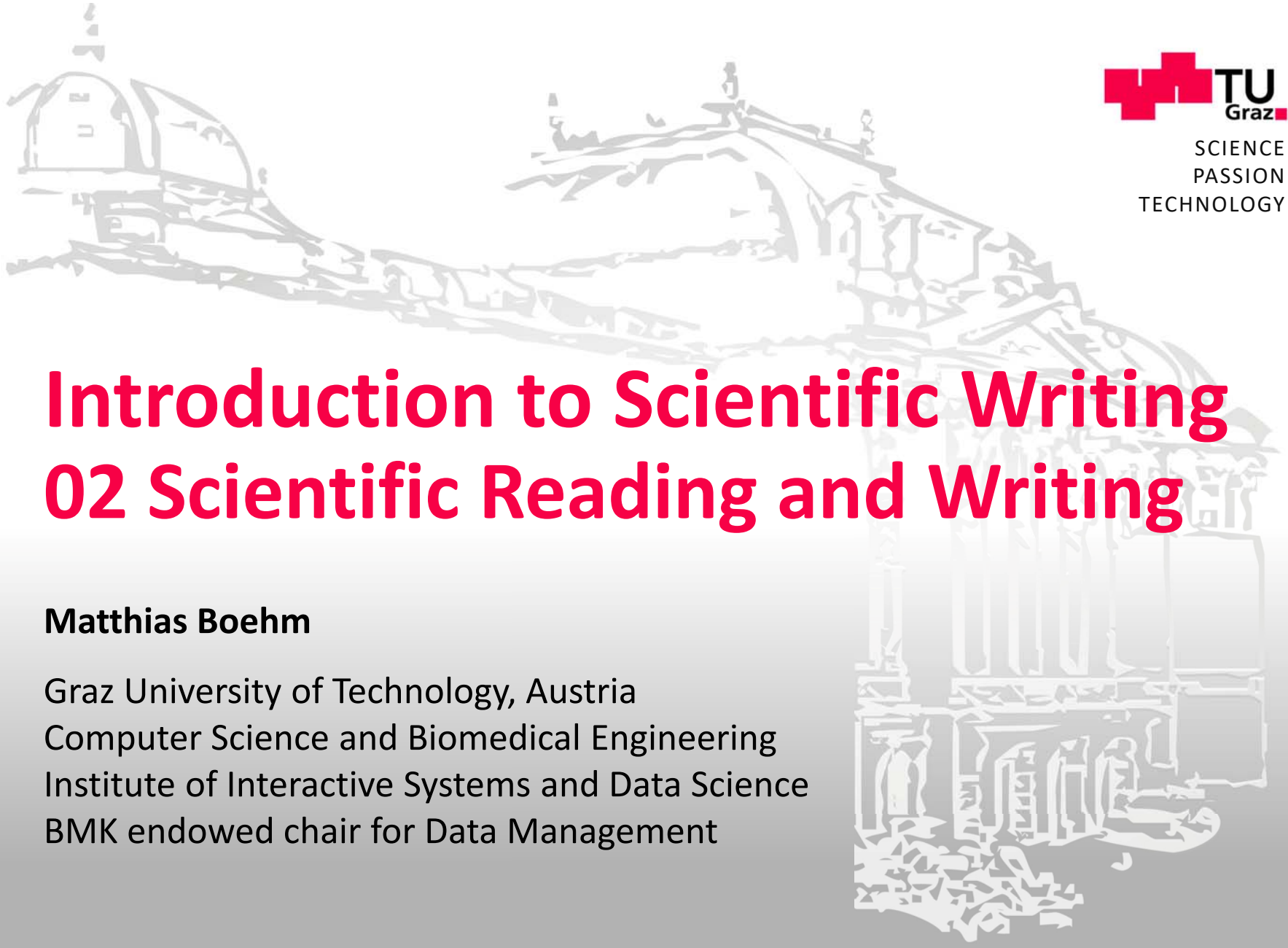


SCIENCE
PASSION
TECHNOLOGY

Introduction to Scientific Writing
02 Scientific Reading and Writing

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: Nov 05, 2020

ISDS

Announcements/Org

- **#1 Virtual Lectures**
  - https://tugraz.webex.com/meet/m.boehm
  - Optional attendance (independent of COVID)

- **#2 Course Registrations** (as of Nov 05)
  - Changes in WS20/21
  - **Introduction to Scientific Writing**

ISDS Group
Boehm
23

Agenda

- **Scientific Reading**
- **Scientific Writing**
- **Paper Project Selection**

Scientific Writing skills can only be learned hands on, and incrementally improve w/ experience

Scientific Reading

In Computer Science (Data Management)

Types of Reading

- **Skimming**
  - **Goal:** understand what the paper/thesis is about
  - Read abstract, and optionally introduction
    scan paper (sections/subsections, structure, figures)

What?

- **Understanding**
  - **Goal:** understand how the presented approach accomplishes the paper's goals
  - **#1** Skimming (see above)
  - **#2** Read the whole paper sequentially, add notes/annotations

How?

- **Reviewing**
  - **Goal:** evaluate potential impact, and limitations
  - **#1** Skimming (see above)
  - **#2** Understanding (see above) + strengths and weaknesses
  - **#3** Write summary, strong/weak points, detailed comments (incl reading)

What's Wrong?

Finding Relevant Work

- **Motivation**
  - Some research areas might be very large (e.g., index structures, compression)
  - How do you find relevant scientific papers/thesis via **multiple channels**
- **By Venue/Year**
  - Start of top-tier conferences/journals and find latest work
  - These papers' related work should provide a good categorization and discussion of related work → **recursive lookup**
- **By Author**
  - Sometimes there are well-known experts in a certain sub-area
  - Find author publications via DBLP and other libraries
- **By Keywords**
  - Broad survey of other related work, to augment the bias year/venue/author approach

Finding Relevant Work, cont.

- **Prefer Trustworthy Sources**
  - Archival publications, awareness of peer-review
  - From right communities (e.g., ML systems vs ML algorithms)
  - Reputation of website, authors, etc

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

Process of Reading – Skimming/Understanding

- **Abstract and Structure**

- **#1 Partial Reading** (mostly skimming)
  - Read into each paragraph until you get what it's about
  - 1st sentence/label: **topic sentence**

- **#2 Fast Reading**
  - Normal reading vs reading w/o vocalization
  - Avoid need for rereading text
    - Back/forward references,
    - Misplacement after distractions
    - Rereading due to lack of understanding

➔ **Read according to your goals of reading**

Process of Reading – Understanding/Evaluation

- **Skepticism**
  - Critical reading is important for **understanding** and **evaluation**
  - **#1** Start open-minded, listen to arguments and trust provided evidence
  - **#2 Don't accept** superficial, contradictory, or unproven claims
  - **#3** If there are problems, which **constructive feedback** could you give or how could the problems be addressed?

- **Questions to Ask Yourself?**
  - What is the problem? Is it a real or artificial problem?
  - How would you solve the problem yourself?
  - How is the paper solving the problem?
  - Is this the simplest approach that yields these results (justified complexity)
  - Are there limitations that are not covered by the paper?
  - Is there existing work that already addresses the same problem?

Reviewing (how NOT to review a paper)

- **Paper Reviews**
  - **Goals:** paper selection, ensure high quality, constructive feedback and recommendations, widen own horizon
  - Lots of similarities to code reviews in OSS

- **Learning by What NOT to Do**
  - Accept if no time to review

[Graham Cormode: How NOT to review a paper: the tools and techniques of the adversarial reviewer. **SIGMOD Rec. 37(4) 2008**]

  - The Goldilocks Method (examples, proofs, theoretical analysis, experiments)
  - If you can't say something nasty … (ignore good parts, focus on weaknesses)
  - Silent but deadly (low scores, no comments)
  - The Natives are Restless (recommend full rewrite by native English speaker)
  - The Referee Moves the Goalpost (changed problem)
  - Blind reviewing

```
This paper leaves many questions unanswered.
      Some claims are questionable.
The paper is of limited interest.
```

Reviewing, cont. (how NOT to review a paper)

- **Introduction**
  - Disagree w/ “Interestingly...”, “Importantly...” or “In practice”,
- **Related Work**
  - “Many important references are omitted”
- **Proposed Method**
  - To simple, impractical, or well-known; correctness?
- **Experiments**
  - Datasets synthetic/real, not all aspects evaluated, too small datasets
- **Conclusions**
  - Disagree w/ every claim; future work can be dismissed

Adversarial Paper Summary

This paper attempts to address the well-studied problem of Graticule Optimization. It proposes the obvious approach of simply storing a set of reference points and calculating offsets. Such approaches are well known in this area. It goes on to propose some simple variations based on precalculating distances. This is an approach that I would expect any straightforward implementation to adopt. Some proof-of-concept experiments show that on a few data sets, the results are slightly better than the most naïve prior methods.

Excursus: An Automatic CS Paper Generator

- **SCIgen: https://pdos.csail.mit.edu/archive/scigen**
  - Generates random CS research papers, incl graphs and figures
  - Uses hand-written context-free grammar
  - Test for low-submission standards of conferences

- **Two Examples**

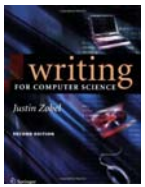
  [Jeremy Stribling, Daniel Aguayo and Maxwell Krohn: Rooter: A Methodology for the Typical Unification of Access Points and Redundancy]
  → accepted as "non-reviewed" **WMSCI 2005**

  [Thomer M. Gil: The Influence of Probabilistic Methodologies on Networking]
  → **rejected**

➔ **Meaningless mix of sentences and technical terms**
(today: GPT-3, what's the take-away for your own papers?)

Scientific Writing

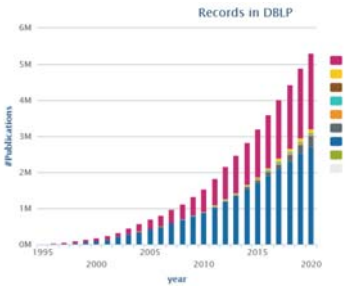
In Computer Science (Data Management)



[Justin Zobel: Writing for Computer Science,
2nd ed. Springer 2004, ISBN 978-1-85233-802-2]

Recap: Writing the Paper

- **#1 Know your Audience**
- **#2 Get your Workflow in Order / Incremental Paper Drafts**
- **#3 Mindset: Quality over Quantity**
  - Aim for top-tier conferences/journals (act as filter)
  - Make the paper useful for others (ideas, evidence, code)



- **#4 Easily Readable: Quality ∝ 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)

Scope and Structure

- **Sections and Subsections**
  - Abstract (see 01 Structure of Scientific Papers)
  - Introduction → context, problem, contributions
  - Background / Preliminaries → necessary background for understanding
  - Main Part → your technical core contributions
  - Main Part 2
  - Experiments → setting, micro benchmarks end-to-end
  - Related Work → areas of related work, differences
  - Conclusions → summary, conclusions, and future work
  - References

- **Recommendations**
  - Avoid sections with only one subsection (e.g., 2 and 2.1)
  - Avoid more than two or at most three nesting levels
  - Clearly separate motivation/background from your own work

Scope and Structure, cont.

- **Bullet Lists**
  - begin{itemize} … \item \end{itemize}
  - begin{enumerate} … \end{enumerate}
- **Figures and Tables**
  - Captions below figures, above tables
- **Code**
  - \begin{verbatim} … \end{verbatim}
- **Theorem, Definition, Examples**
  - Refine theorem environments as needed
- **Algorithms**
  - Can be clearer than text, but not always
  - Carefully select the right level of abstraction

Data Structure: The MNC sketch $\mathbf{h}_A$ of an $m \times n$ matrix $\mathbf{A}$ comprises the following information, where we use $\mathbf{h}$ as a shorthand whenever the context is clear.

- *Row/Column NNZs:* Count vectors $\mathbf{h}^r = \text{rowSums}(\mathbf{A} \neq 0)$ and $\mathbf{h}^c = \text{colSums}(\mathbf{A} \neq 0)$ indicate the NNZs per row and column, where $\mathbf{h}^r_i$ is the count of the ith row.
- *Extended Row/Column NNZs:* Count vectors $\mathbf{h}^{er} = \text{rowSums}((\mathbf{A} \neq 0) \cdot (\mathbf{h}^c = 1))$ and $\mathbf{h}^{ec} = \text{colSums}((\mathbf{A} \neq 0) \cdot (\mathbf{h}^r = 1))$ indicate the NNZs per row/column that appear in columns/rows with a single non-zero.

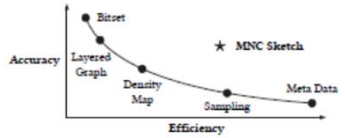


Figure 2: Accuracy/Efficiency Goal of the MNC Sketch.

Table 1: Analysis of Existing Sparsity Estimators.

Estimator	Space	Time	%	Bias
MetaAC E_{ac}	$O(1)$	$O(1)$	✓	
MetaWC E_{wc}	$O(1)$	$O(1)$	✓	sc
Bitset E_{bmm}	$O(mn + nl + ml)$	$O(mnl)$	✓	
DMap E_{dm}	$O(\frac{mn+nl+ml}{b^2})$	$O(\frac{mnl}{b^2})$	✓	
Sample E_{smpl}	$O(\lvert S \rvert)$	$O(\lvert S \rvert(m + l))$		sc
LGraph E_{sph}	$O(rd + \text{nnz}(A, B))$	$O(r(d + \text{nnz}(A, B)))$	✓	
MNC E_{mnc}	$O(d)$	$O(d + \text{nnz}(A, B))$	✓	

THEOREM 3.1. *Given MNC sketches $\mathbf{h}_A$ and $\mathbf{h}_B$ for matrices $\mathbf{A}$ and $\mathbf{B}$, the output sparsity s_C of the matrix product $\mathbf{C} = \mathbf{A}\,\mathbf{B}$ can be exactly computed under the assumptions A1 and A2 via a dot product of $\mathbf{h}^c_A$ and $\mathbf{h}^r_B$:*

$$s_C \equiv \hat{s}_C = \mathbf{h}^c_A \mathbf{h}^r_B / (ml) \text{ if } \max(\mathbf{h}^r_A) \leq 1 \vee \max(\mathbf{h}^c_B) \leq 1. \quad (7)$$

Algorithm 1 MNC Sparsity Estimation

Input: MNC sketches $\mathbf{h}_A$ and $\mathbf{h}_B$ for matrices $\mathbf{A}$ and $\mathbf{B}$
Output: Output sparsity s_C

```
1: // a) basic and extended sparsity estimation, incl upper bound
2: if max(h^r_A) ≤ 1 ∨ max(h^c_B) ≤ 1 then          // see Theorem 3.1
3:     nnz ← h^c_A h^r_B
4: else if exists(h^{er}_A) ∨ exists(h^{er}_B) then // extended NNZ counts
5:     nnz ← h^{ec}_A h^r_B + (h^c_A − h^{ec}_A) h^{er}_B   // exact fraction
6:     p ← (nnz(h^r_A) − |h^r_A = 1|) · (nnz(h^c_B) − |h^c_B = 1|)   // #cells
7:     nnz ← nnz + E_dm(h^c_A − h^{ec}_A, h^r_B − h^{er}_B, p) · p   // generic rest
8: else                                             // generic fallback estimate
9:     p ← nnz(h^r_A) · nnz(h^c_B)                  // #cells
10:    nnz ← E_dm(h^c_A, h^r_B, p) · p
11: // b) apply lower bound, see Theorem 3.2
12: nnz ← max(nnz, |h^r_A > n/2| · |h^c_B > n/2|)    // lower bound
13: return s_C ← nnz/(ml)
```

Formatting

- **Motivation**
  - A carelessly formatted paper (layout, figures, fonts, underlining) creates a bad first impression
  - Recap: **skimming** and **anchoring**

"The paper's approach is probably equally sloppy"

- **Figures**
  - Use same font and font size as the main text / code in main paper
  - Avoid text overlap, too aggressive **colors**

- **Orphans and Widows**
  - Imprecise definition
  - Avoid few words per line, single line at next page

Strength Reduction: Note that cumsumprod($\mathbf{X}$) uses cumsumprod($\mathbf{B}$)$_{n1}$—i.e., the last block entry—as part of f_{agg}. Similarly, for cumsum($\mathbf{X}$), we could use cumsum($\mathbf{B}$)$_{n:}$. However, this simplifies to colSums($\mathbf{B}$), which avoids materializing the cumsum output block.

Looks ugly and wastes lots of space

- **Highlighting**
  - Use \emph{} (emphasize) over underlining or bold

Punctuation

- **Commas**
  - Whenever a pause is appropriate, or required to avoid ambiguity

```
When using disk[,] tree algorithms
were found to be particularly poor.

A woman without her man is nothing.
A woman: without her, man is nothing.
```

  - **Lists:** red, blue, black, and white (oxford/serial comma)
  - **Special sentence start:** However, Hence, Therefore, In this paper,

- **Semicolons**
  - Divide a long sentence into sub-sentences, or separation for emphasis
  - Lists with sublists

```
We use index structures like b-trees, tries,
and hash tables; as well as compression
techniques like run-length encoding,
dictionary encoding, and null suppression.
```

- **Exclamations**
  - Avoid exclamation marks! Never use more than one!!

Writing Style

- **Goal: Clear, easy-to-read writing**
- **Variation**
  - Diversity (structure, length of sentences/paragraphs, choice of words, sentence beginning) helps keeping the reader's attention

```
The system of rational numbers is incomplete. This was
discovered 2000 years ago by the Greeks. The problem
arises in squares with sides of unit length. The length
of the diagonals of these squares is irrational. This
discovery was a serious blow to the Greek mathematicians.
```

```
The Greeks discovered 2000 years ago that the system of
rational numbers is incomplete. The problem is that some
quantities, such as the length of the diagonal of a
square with unit sides, are irrational. This discovery
was a serious blow to the Greek mathematicians.
```

Writing Style, cont.

- **Prefer Active Voice**
  - Easier to understand, shorter, more interesting to read
  - Use we over I

```
In this section, the background and motivation
for compressed linear algebra is introduced.
```

```
In this section, we provide the background and
motivation for compressed linear algebra.
```

- **Prefer Present Tense**
  - Most content of a research paper can be described in present
  - Exceptions: user studies, (specific experimental setup)

- **Use of References**
  - Use \cite{key1,key2}
  - **Don't use refs as nouns**
  - **Prefer primary sources**

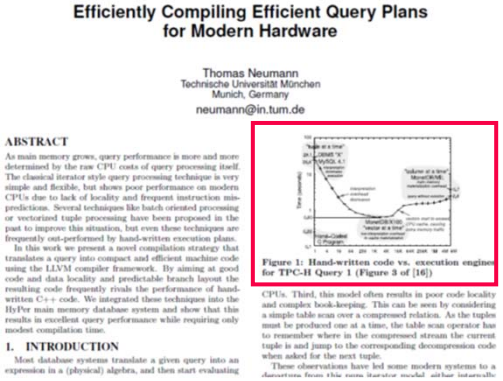
```
Later, [40] investigated query
processing on heavyweight Huffman
coding schemes,
```

```
Later, Raman and Swart investigated
query processing on heavyweight
Huffman coding schemes [40],
```

Plagiarism

- **#1 Self-Plagiarism (Bad Idea)**
  - Avoid reusing motivation, introduction, figures, and examples
  - Start writing every thesis / paper from scratch (unless thesis summaries/extends previous papers)

- **#2 Figure Plagiarism (Bad Idea)**
  - Never copy figures from other papers, web, etc
  - Create all figures yourself, even for surveys (can be based on ideas of existing papers)
  - Exceptions do exist w/ explicit references



Efficiently Compiling Efficient Query Plans for Modern Hardware

Thomas Neumann
Technische Universität München
Munich, Germany
neumann@in.tum.de

ABSTRACT

As main memory grows, query performance is more and more determined by the raw CPU costs of query processing itself. The classical iterator style query processing technique is very simple and flexible, but shows poor performance on modern CPUs due to lack of locality and frequent instruction mispredictions. Several techniques like batch oriented processing or vectorized tuple processing have been proposed in the past to improve this situation, but even these techniques are frequently out-performed by hand-written execution plans.

In this work we present a novel compilation strategy that translates a query into compact and efficient machine code using the LLVM compiler framework. By aiming at good code and data locality and predictable branch layout the resulting code frequently rivals the performance of hand-written C++ code. We integrated these techniques into the HyPer main memory database system and show that this results in excellent query performance while requiring only modest compilation time.

1. INTRODUCTION

Most database systems translate a given query into an expression in a (physical) algebra, and then start evaluating

Figure 1: Hand-written code vs. execution engines for TPC-H Query 1 (Figure 3 of [16])

CPUs. Third, this model often results in poor code locality and complex book-keeping. This can be seen by considering a simple table scan over a compressed relation. As the tuples must be produced one at a time, the table scan operator has to remember where in the compressed stream the current tuple is and jump to the corresponding decompression code when asked for the next tuple.

These observations have led some modern systems to a

- **#3 Plagiarisms (Really Bad Idea)**
  - Never copy figures or text from other peoples work and claim its yours (slight rewording does not change that)
  - For archival scientific publications, there is a high chance it will be detected

Plagiarism – Duplicate Submission

- **Example SIGMOD'21**

A research paper submitted to SIGMOD 2021 cannot be under review for any other publishing forum or presentation venue, including conferences, workshops, and journals, during the time it is being considered for SIGMOD. Furthermore, after you submit a research paper to SIGMOD, you must await the response from SIGMOD and only re-submit elsewhere if your paper is rejected - or withdrawn at your request - from SIGMOD. This restriction applies not only to identical papers but also to papers with a substantial overlap in scientific content and results.

Every research paper submitted to SIGMOD 2021 must present substantial novel research not described in any prior publication. In this context, a prior publication is (a) a paper of five pages or more presented, or accepted for presentation, at a refereed conference or workshop with proceedings; or (b) an article published, or accepted for publication, in a refereed journal. If a SIGMOD 2021 submission has overlap with a prior publication, the submission must cite the prior publication, along with all other relevant published work, following the guidelines in the Anonymity Requirements for Double-Blind Reviewing section below.

Page Limits

[Credit: https://twitter.com/fadeladib/status/1322646406088347649]

- **Most Conferences/Journals**
  - Given predefined template, changes not permitted
  - SIGMOD/PVDLB: **12 pages + unlimited references**
  - ICDE: 12 pages incl. references
- **#1 Avoid Cheating**
  - Don't change the template, fonts, or margins (at least not too excessively)
  - Condensing more text into the paper will make it harder to read
- **#2 Carefully Trim Down Draft**
  - Write unlimited paper, then select, and revise
  - Write and revise section by section as you write

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

- **#3 Never Excuse Missing Content by "lack of space"**

```
Due to the lack of space, we omit
[essential details] /
[essential experiments]
```

Paper Projects

In Computer Science (Data Management)

Overview 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**~~ **Nov 05:** List of projects proposals, feel free to bring your own, or ask for extended proposals (e.g., ML systems, distributed systems)
  - **Nov 12:** project selection via email to m.boehm@tugraz.at (11.59pm) subject: [Scientific Writing] Project Selection
  - **Dec 23:** paper submission via email to m.boehm@tugraz.at (11.59pm)
  - **Jan 07:** Final project presentation (all students)

Paper Projects

- **Visual Analytics**
  - **#1** Tarique Siddiqui, Paul Luh, Zesheng Wang, Karrie Karahalios, Aditya G. Parameswaran: ShapeSearch: A Flexible and Efficient System for Shape-based Exploration of Trendlines. SIGMOD 2020
  - **#2** Uwe Jugel, Zbigniew Jerzak, Gregor Hackenbroich, Volker Markl: M4: A Visualization-Oriented Time Series Data Aggregation. PVLDB 7(10) 2014

- **Video Analytics**
  - **#3** Daniel Kang, Peter Bailis, Matei Zaharia: BlazeIt: Optimizing Declarative Aggregation and Limit Queries for Neural Network-Based Video Analytics. PVLDB 13(4) 2019
  - **#4** Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, Matei Zaharia: NoScope: Optimizing Deep CNN-Based Queries over Video Streams at Scale. PVLDB 10(11) 2017

Paper Projects, cont.

- **Fairness / Causality / Diversity**
  - **#5** Julia Stoyanovich, Bill Howe, H. V. Jagadish: Responsible Data Management. PVLDB 13(12) 2020
  - **#6** Babak Salimi, Luke Rodriguez, Bill Howe, Dan Suciu: Interventional Fairness: Causal Database Repair for Algorithmic Fairness. SIGMOD 2019
  - **#7** Marina Drosou, Evaggelia Pitoura: DisC diversity: result diversification based on dissimilarity and coverage. PVLDB 6(1) 2012

- **Graphs**
  - **#8** Siddhartha Sahu, Amine Mhedhbi, Semih Salihoglu, Jimmy Lin, M. Tamer Özsu: The Ubiquity of Large Graphs and Surprising Challenges of Graph Processing. PVLDB 11(4) 2017
  - **#9** Yuanyuan Tian, Andrey Balmin, Severin Andreas Corsten, Shirish Tatikonda, John McPherson: From "Think Like a Vertex" to "Think Like a Graph". PVLDB 7(3) 2013
  - **#10** Grzegorz Malewicz, Matthew H. Austern, Aart J. C. Bik, James C. Dehnert, Ilan Horn, Naty Leiser, Grzegorz Czajkowski: Pregel: a system for large-scale graph processing. SIGMOD 2010

Paper Projects, cont.

- **NLP**
  - **#11** Diptikalyan Saha, Avrilia Floratou, Karthik Sankaranarayanan, Umar Farooq Minhas, Ashish R. Mittal, Fatma Özcan: ATHENA: An Ontology-Driven System for Natural Language Querying over Relational Data Stores. PVLDB 9(12) 2016
  - **#12** Fei Li, H. V. Jagadish: Constructing an Interactive Natural Language Interface for Relational Databases. PVLDB 8(1) 2014

- **Provenance**
  - **#13** Pingcheng Ruan, Gang Chen, Anh Dinh, Qian Lin, Beng Chin Ooi, Meihui Zhang: Fine-Grained, Secure and Efficient Data Provenance for Blockchain. PVLDB 12(9) 2019
  - **#14** Daniel Deutch, Nave Frost, Amir Gilad: Provenance for Natural Language Queries. PVLDB 10(5) 2017
  - **#15** Adriane Chapman, H. V. Jagadish: Why not? SIGMOD 2009

Paper Projects, cont.

- **Query Compilation**
  - **#16** Timo Kersten, Viktor Leis, Alfons Kemper, Thomas Neumann, Andrew Pavlo, Peter A. Boncz: Everything You Always Wanted to Know About Compiled and Vectorized Queries But Were Afraid to Ask. PVLDB 11(13) 2018
  - **#17** Prashanth Menon, Andrew Pavlo, Todd C. Mowry: Relaxed Operator Fusion for In-Memory Databases: Making Compilation, Vectorization, and Prefetching Work Together At Last. PVLDB 11(1) 2017
  - **#18** Andrew Crotty, Alex Galakatos, Kayhan Dursun, Tim Kraska, Carsten Binnig, Ugur Çetintemel, Stan Zdonik: An Architecture for Compiling UDF-centric Workflows. PVLDB 8(12) 2015
  - **#19** Milos Nikolic, Mohammed Elseidy, Christoph Koch: LINVIEW: incremental view maintenance for complex analytical queries. SIGMOD 2014
  - **#20** Yannis Klonatos, Christoph Koch, Tiark Rompf, Hassan Chafi: Building Efficient Query Engines in a High-Level Language. PVLDB 7(10) 2014
  - **#21** Thomas Neumann: Efficiently Compiling Efficient Query Plans for Modern Hardware. Proc. VLDB Endow. 4(9) 2011

Paper Projects, cont.

- **Reusing Intermediates**
  - **#22** Doris Xin, Litian Ma, Jialin Liu, Stephen Macke, Shuchen Song, Aditya G. Parameswaran: Helix: Accelerating Human-in-the-loop Machine Learning. PVDLB 11(12) 2018
  - **#23** Ce Zhang, Arun Kumar, Christopher Ré: Materialization optimizations for feature selection workloads. SIGMOD 2014
  - **#24** Milena Ivanova, Martin L. Kersten, Niels J. Nes, Romulo Goncalves: An architecture for recycling intermediates in a column-store. SIGMOD 2009

- **Raw Query Processing**
  - **#25** Manos Karpathiotakis, Ioannis Alagiannis, Anastasia Ailamaki: Fast Queries Over Heterogeneous Data Through Engine Customization. PVLDB 9(12) 2016
  - **#26** Ioannis Alagiannis, Renata Borovica, Miguel Branco, Stratos Idreos, Anastasia Ailamaki: NoDB: efficient query execution on raw data files. SIGMOD 2012

Paper Projects, cont.

- **Transactions / High Availability**
  - **#27** Yihe Huang, William Qian, Eddie Kohler, Barbara Liskov, Liuba Shrira: Opportunities for Optimism in Contended Main-Memory Multicore Transactions. PVLDB 13(5) 2020
  - **#28** Umar Farooq Minhas, Shriram Rajagopalan, Brendan Cully, Ashraf Aboulnaga, Kenneth Salem, Andrew Warfield: RemusDB: Transparent High Availability for Database Systems. PVLDB 4(11) 2011
  - **#29** Michael J. Cahill, Uwe Röhm, Alan D. Fekete: Serializable isolation for snapshot databases. SIGMOD 2008
- **Data Flow Systems**
  - **#30** Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauly, Michael J. Franklin, Scott Shenker, Ion Stoica: Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing. NSDI 2012
  - **#31** Christopher Olston, Shubham Chopra, Utkarsh Srivastava: Generating example data for dataflow programs. SIGMOD 2009

Paper Projects, cont.

- **Database Cracking**
  - **#32** Pedro Holanda, Stefan Manegold, Hannes Mühleisen, Mark Raasveldt: Progressive Indexes: Indexing for Interactive Data Analysis. PVLDB 12(13) 2019
  - **#33** Felix Martin Schuhknecht, Alekh Jindal, Jens Dittrich: The Uncracked Pieces in Database Cracking. PVLDB 7(2) 2013
  - **#34** Stratos Idreos, Martin L. Kersten, Stefan Manegold: Database Cracking. CIDR 2007
- **Index Structures**
  - **#35** Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, Neoklis Polyzotis: The Case for Learned Index Structures. SIGMOD 2018
  - **#36** Huanchen Zhang, Hyeontaek Lim, Viktor Leis, David G. Andersen, Michael Kaminsky, Kimberly Keeton, Andrew Pavlo: SuRF: Practical Range Query Filtering with Fast Succinct Tries. SIGMOD 2018
  - **#37** Viktor Leis, Alfons Kemper, Thomas Neumann: The adaptive radix tree: ARTful indexing for main-memory databases. ICDE 2013
  - **#38** Nicolas Bruno, Surajit Chaudhuri: Constrained physical design tuning. Proc. VLDB Endow. 1(1) 2008

Paper Projects, cont.

- **Compression**
  - **#39** Peter A. Boncz, Thomas Neumann, Viktor Leis: FSST: Fast Random Access String Compression. Proc. VLDB Endow. 13(11) 2020
  - **#40** Daniel J. Abadi, Samuel Madden, Miguel Ferreira: Integrating compression and execution in column-oriented database systems. SIGMOD 2006
  - **#41** Marcin Zukowski, Sándor Héman, Niels Nes, Peter A. Boncz: Super-Scalar RAM-CPU Cache Compression. ICDE 2006
- **Modern Hardware**
  - **#42** Clemens Lutz, Sebastian Breß, Steffen Zeuch, Tilmann Rabl, Volker Markl: Pump Up the Volume: Processing Large Data on GPUs with Fast Interconnects. SIGMOD 2020
  - **#43** Ismail Oukid, Johan Lasperas, Anisoara Nica, Thomas Willhalm, Wolfgang Lehner: FPTree: A Hybrid SCM-DRAM Persistent and Concurrent B-Tree for Storage Class Memory. SIGMOD 2016
  - **#44** Changkyu Kim, Jatin Chhugani, Nadathur Satish, Eric Sedlar, Anthony D. Nguyen, Tim Kaldewey, Victor W. Lee, Scott A. Brandt, Pradeep Dubey: FAST: fast architecture sensitive tree search on modern CPUs and GPUs. SIGMOD 2010
  - **#45** René Müller, Jens Teubner, Gustavo Alonso: Data Processing on FPGAs. Proc. VLDB Endow. 2(1) 2009

Summary and Q&A

- **Scientific Reading**
- **Scientific Writing**
- **Paper Project Selection**

- **Remaining Questions?**

- **Next Lectures**
  - 03 Experiments, Reproducibility, and Projects [Nov 12]