Scientific Project: Databases for Multi-dimensional Data, Genomics and modern Hardware

David Broneske, Gabriel Campero, Bala Gurumurthy, Marcus Pinnecke, Gunter Saake

, October 18, 2019
Overview

- Concepts of this course
- Course of action (milestones, presentations)
- Overview of project topics & forming project teams
- How to perform literature research?
Overview

- Concepts of this course
- Course of action (milestones, presentations)
- Overview of project topics & forming project teams
- How to perform literature research?
- Further lectures:
  - Academic writing (2 lectures)
Organization
Scientific Project: Modules

**Bachelor**

- **Module:** WPF FIN SMK (Schlüssel- und Methodenkompetenzen)
- 5 CP = 150h ⇒ 42h presence time (3 SWS) + 108h autonomous work

**Master**

- **Module:** Scientific Team Project (Inf, IngInf, WIF, CV)
  - DKE: Methods 2 or Applications
  - DE: Interdisciplinary Team Project, Specialization
- 6 CP = 180h ⇒ 42h presence time (3 SWS) + 138h autonomous work

*Grade at the end of the course for the whole project team*
Scientific Project: Prerequisite

- Successful programming test in C++/Java/Python
- 1h theoretical test in a seminar room (data and place to be discussed)
- Half of the team members have to pass the test
- Topics:
  - Some language specifics
  - General program understanding
  - Control flow understanding
- You can take all tests and have to pass at least one!
<table>
<thead>
<tr>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>04.11.2019</td>
<td>05.11.2019</td>
<td>06.11.2019</td>
<td>07.11.2019</td>
</tr>
<tr>
<td><strong>Winter Break</strong></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td><strong>MS III</strong></td>
<td>06.01.2020</td>
<td>07.01.2020</td>
<td>08.01.2020</td>
<td>09.01.2020</td>
</tr>
<tr>
<td><strong>MS Final</strong></td>
<td>27.01.2020</td>
<td>28.01.2020</td>
<td>29.01.2020</td>
<td>30.01.2020</td>
</tr>
</tbody>
</table>
Milestones:

- **Milestone I** - Topic, schedule, and team presentation & first results of literature research
- **Milestone II** - Concept & additional literature research
- **Milestone III** - Implementation & evaluation setup
- **Milestone IV** - Final presentation (wrap-up + evaluation results)
Concepts & Content
Lecture, Meetings & Presentation

*Lecture & Presentation*

- Time/Place: Friday, 13:00-15:00, G22A - 208
- Lectures with content of course → all
- Presentation of *main milestones* (see time table)
  → each project team

*Meetings (Exercise)*

- Individual for each project team
- Time and room to be agreed in project teams!
- Presentation of all intermediate results/milestones (informal)
- Discussion, discussion, discussion . . .
Progress of Course

Deliveries

- 4 milestone presentations (main milestones)
- Each team member has to present at least once
- Reporting of (sub) milestones in exercises/meetings
- Written paper about literature research (technical report)
- Prototypical implementation
Deliveries and Grading (I)

*Technical Report*

- Delivery of report at a given time (deadline)
- Quality/Quantity of literature research
- Number of pages
- Quality of paper structure and evaluation
- Own contribution
Deliveries and Grading (II)

*Presentation & Discussion*

- Quality of scientific presentation (structure, references, time)
- Assessment regarding the content (e.g., results of particular milestones)
- Participation of discussion

*Organization*

- Strictness
- Communication (just-in-time answers, satisfying time constraints)
- Self-organization (Sharing tasks, internal reporting of current state-of-work, dealing with problems)
- Autonomous working
Deliveries and Grading (III)

- Grade consists of:
  - Presentations: 30%,
  - Implementation: 30%,
  - Paper: 30%,
  - Soft Skills: 10%

- Binding registration: Second Milestone
Objectives & Qualification (I)

Acquired skills, specific to research

- Performing literature research
- Understanding and structured reviewing of scientific work
- Autonomous, solution-based reasoning on research task (e.g., finding alternative solutions)
- How to ask? How to adapt a task (extend/reduce)?
- Academic writing
Objectives & Qualification (II)

Acquired skills, always needed

- Team management
- Project and time scheduling
- Presentation of results
- Flexibility regarding changing conditions
- Reasoning about solutions ("Why is this the best/not adequate...")
Task & Time Management

Task Management

- **Main milestones** have to be finished in time
- **(Sub) milestones** are less strict (but don’t be sloppy)
- Pre-defined work packages ⇒ each project team
  - ...defines sub work packages
  - ...determines responsibilities for these packages (divide & conquer)

Time Management

- Planning of periods
- Regarding capacities and resources
- Considering other tasks and activities
- Reporting of delays immediately to project members!
Role Management

• Possible roles: team leader, developer, researcher, ...

• work together vs. responsibilities: design, implementation, testing, writing, ...

• Delegate for important roles/work packages

• Assignment of (sub) tasks to role for each milestone
Topic & Project Teams

- Teams with 4 to 6 students
- Most tasks can be chosen once

Projects

- Theoretical part
  - State of the art
  - New ideas

- Practical part
  - Usually in C++, Java, or Python
  - Prototypical implementation
  - Evaluation part
Intro: Data skipping in Hadoop

- A good data fragmentation is necessary for scalable processing in Big Data.

- **Aggressive data skipping** is the SOTA ML approach, using hierarchical agglomerative clustering with Ward’s method.

- We developed a competitor based on deep reinforcement learning. How good is our solution? (can we make it better?)

Your Task

- Literature research: aggressive data skipping, basics on deep reinforcement learning.

- Prototypical implementation of aggressive data skipping using Spark. Experimental evaluation and analysis comparing with our DRL solution.

- Long-term goal: A production-ready open source AI-based partitioning tool & a paper for consideration in SysML.
Topic 2 - Similarity Skipper (Sim-Skip)

**Intro: Learning to hash for high dimensional data management**

- Top-k search in dense high-dimensional vectors requires specialized solutions. This is a very relevant application.

- When data is large, even parallel scans will be inefficient.

- Learned hashing is the current SOTA for managing such kind of data in image domains. How good does this technique perform on structured relational data?

**Your Task**

- Literature research: Hadoop file formats, deep hashing, triplet training of neural networks.

- Prototypical implementation of a deep hashing process using Tensorflow and Spark. Experimental evaluation and analysis.

- Long-term goal: A workshop paper in DEEM@SIGMOD.
Intro: Applications of deep learning for graph data processing

- Graphs are everywhere & graph technologies are a moving target. To date still very heuristic-driven. ML is barely tested.
- Differential neural computers are already able to act like primitive graph query engines.
- How good do models like these fare nowadays for graph processing, and what needs to be improved?

Your Task

- Literature research: differential neural computers, graph nets, RDF3x.
- Prototypical implementation using existing libraries from Deepmind, selected datasets. Experimental evaluation and analysis.
- Long-term goal: A workshop paper in GRADES/AIDM@SIGMOD, or AIDB@VLDB.
Intro

- Heterogeneous hardware used for improving performance
- Work partitioning is problematic and requires training data for best device selection
- Further, data parallel execution cannot be decided in prior

We’ve got

- Dispatcher implementation
- Iterative functional parallel executor

Your Task

- Literature Research: data-parallel, iterative and cross-device execution systems
- Understanding of morsel driven parallelism
- Invention of a clever concept to perform data as well as functional parallel execution across devices
- Implementation of your concept for missing database operators - hash join
- Benchmarking the execution system
Intro

- Query processing models for GPU: block-at-a-time & compiled execution
- Block-at-a-time: execute data bulk/function after next in query pipeline
- Compiled: generates execution code in runtime and execute them together
- Not sure which execution model is most suitable for GPUs

We’ve got

- GPU accelerated DBMS - CoGaDB
- Functionalities for b-a-a-t and compiled execution
- Concepts for evaluating of models in stand-alone CPU

Your Task

- Literature Research: Execution models for heterogeneous hardware
- Understanding execution of CoGaDB
- Developing evaluation suite for GPUs
- Implementation missing operations and the evaluation suite constructs
- Critical analysis on the results and investigation on the resultant values
Intro

- Management of semi-structured data (e.g., JSON) has become daily business
- A widespread of formats for physical storage of JSON-like data exists today (e.g., PlainText, BSON, CBor, Avro, Parquet, FlatBuffers, MessagePack, Smile, UBJSON, Ion, Carbon)
- A comprehensive comparison about design goals, concrete data model, limits and benefits, as well as operators, on an abstract, theoretical level is missing, though

We’ve got

- Carbon (Columnar Binary Json) implementation and specification as starting point
- Running theses on evaluation of Carbon vs BSON

Your Task

- Literature Research: Semi-Structured Data Model (math model for JSON-like data), and existing evaluations for formats mentioned above → understanding on an abstract level
- Research for existing comparisons, design goals, and applications of models above
- Creation of common theoretical model and common taxonomy for formats above
- List missing evaluations, top 5 formats, and come up with at least 5 further advanced and reasonable metrics not considered so far
- Implementation of an evaluation framework with new metrics and missing evaluations as proof of concepts for at least 5 formats (PlainText, BSON, UBJSON, and Carbon excluded)
Topic 7 - Order-By Queries in Elf

Intro

• Elf: multi-dimensional main memory index structure for efficient selections
• Stores data sorting in a multi-dimensional order
• Common data-intensive operator: Sorting

We’ve got

• Elf implementation in C++

Your Task

• Literature Research: Related index structures and sorting algorithms
• Understanding of the Elf and its optimization concepts
• Implementation Sorting Operator for Elf
• Performance evaluation against sequential scans
Accelerating multi-column selection predicates in Online Analytical Processing (OLAP) scenarios where single predicate selectivities are above the threshold of 2%. For the TPC-H Query, a full table scan is even smaller than for disk-based databases. This task has become even more complex, increasing the number of evaluated selection predicates per query and table. Recent approaches to increase performance benefits from several factors, such as the Elf tree structure that is able to exploit the relation between data of several columns. Elf results indicate a clear superiority of our approach for multi-column selection. Prefix-redundancy elimination reduces intermediate results, which are the input for predicate evaluation. In this paper, we examine the question of how to achieve a speed-up between a factor of five and two orders of magnitude when using accelerated scans. However, scanning each column one by one leaves tuning opportunities open that arise if all predicates are to be indexed. In performance benefits from several factors up to two orders of magnitude, we find that the accumulated selectivity of the multi-column selection predicate is an important task in current OLAP scenarios.
Topic 8 - Parallel Build of Elf

Intro

• Elf: multi-dimensional main memory index structure for efficient selections
• Stores data sorting in a multi-dimensional order
• Building involves a multi-dimensional sort → expensive

We’ve got

• Elf implementation in C++

Your Task

• Literature Research: Parallelization strategies for building index structures
• Understanding of the Elf and its optimization concepts
• Implementation different parallelization strategies for Elf
• Performance evaluation against standard implementation
Topic 9 - GPU-Accelerated Selections in Elf

*Intro*

- Elf: multi-dimensional main memory index structure for efficient selections
- Stores data sorting in a multi-dimensional order
- Multi-core architectures demand for a clever parallelization strategy for GPUs

*We’ve got*

- Elf implementation in C++
- Parallel search and insert for CPU

*Your Task*

- Literature Research: Related parallelization strategies for index structures
- Understanding of the Elf and its optimization concepts
- Implementation GPU traversal variants for Elf
- Performance evaluation against serial/CPU-parallel implementation
Finding your Team

Topics:

• Topic 1 - Fragment Skipper (Gridformation)
• Topic 2 - Deep hashing (Sim-Skip)
• Topic 3 - Graph processing in a neural network (Malko)
• Topic 4 - Cross-device data parallel query processing
• Topic 5 - Evaluating GPU-based Execution Models
• Topic 6 - Indepth Semi-Structured Data Storage
• Topic 7 - Order-By Queries on Elf
• Topic 8 - Parallel Build of Elf
• Topic 9 - GPU-Accelerated Selections in Elf

When do we meet for the programming test?
Literature Research
How to Perform Literature Research

- Efficient literature research requires
  - Knowledge of *Where* to search
  - Knowledge of *How* to search
  - Finding adequate search terms
  - Structured review of papers
  - Knowledge of how to find information in papers
Where to Search (I)

- Different websites available that provide large literature databases

   - Key word and concrete paper search
   - Often, PDFs are provided

2. DBLP: [http://www.informatik.uni-trier.de/~ley/db/](http://www.informatik.uni-trier.de/~ley/db/)
   - Search for keyword, conferences, journals, author(s)
   - BibTex and references to other websites

3. Citeseer: [http://citeseerx.ist.psu.edu/about/site](http://citeseerx.ist.psu.edu/about/site)
   - keyword, fulltext, author, and title search
   - BibTex and (partially) PDFs are provided
Where to Search (II)

• Publisher sites are also a suitable target

• ACM Digital Library: http://portal.acm.org/dl.cfm
  ■ Keyword, author, conference/literature (proceedings), and title search
  ■ Bibtex, mostly PDFs and other information are provided

• IEEE Xplore: http://ieeexplore.ieee.org/Xplore/guesthome.jsp?reload=true
  ■ Similar to ACM, but only few PDFs
  ■ Extended access within university network

• Springer: http://www.springerlink.de/
  ■ Similar to previous
  ■ Extended access within university Network

• Further search possibilities: on author, research group or university sites
How to Search

Some hints to not get lost in the jungle

• Use distinct keywords (fingerprint vs. fingerprint data)

• Keep keywords simple (at most three words)

• Otherwise, search for whole title

• Read abstract (and maybe introduction) ⇒ decision for relevance

First insights

• Read abstract, introduction and background/related work (coarse-grained) to
  ■ ... get a first idea of the approach
  ■ ... find other relevant papers
Finding the required information

- Read the paper carefully
- Omit formal parts/sections
- Try to classify (core idea, main characteristics) \(\Rightarrow\) develop classification/evaluation in mind
- Understand the big picture
- Make notes
- Do NOT translate each sentence
Finding your Team

Topics:

• Topic 1 - Fragment Skipper (Gridformation)
• Topic 2 - Deep hashing (Sim-Skip)
• Topic 3 - Graph processing in a neural network (Malko)
• Topic 4 - Cross-device data parallel query processing
• Topic 5 - Evaluating GPU-based Execution Models
• Topic 6 - Indepth Semi-Structured Data Storage
• Topic 7 - Order-By Queries on Elf
• Topic 8 - Parallel Build of Elf
• Topic 9 - GPU-Accelerated Selections in Elf