Advanced Topics in Databases

Introduction

David Broneske, Gabriel Campero, Bala Gurumurthy, Andreas Meister, Marcus Pinnecke, Gunter Saake, Roman Zoun
Otto-von-Guericke University Magdeburg
Summer Term 2018
Motivation for the Lecture

- Familiarize students with current developments in database research

- Topics chosen:
  - First solutions currently making their way into database management systems and applications → practical relevance
  - Solutions not yet fully developed and open problems exist → research relevance

- Possible starting points for scientific work, e.g. master thesis, position in academia, Ph.D. thesis, industry R&D, etc.
Yesterday’s DBMS Landscape

- Oracle
- IBM DB2
- SQL Server
- Teradata

Application

DBMS

Database

Application

„Banking, SAP, …“

„Server“

„Disk“
Yesterday’s DBMS Hardware

Small main memory

Disk-based systems
Assumptions of yesterday’s DBMSs

- Capacity of main memory <1% of the stored data
- Fixed block size based on the transfer unit between disks and main memory
- Central scheduler to schedule transactions
- No redundant data storage in main memory
- Pipelining is always beneficial (no storage of intermediate results)
- Compiling of SQL for one processor architecture → Reuse of compiled plan
Today’s DBMS Topics

Genomic Data In Memory

Cloud Co Processing

New Storage Devices

Parallel Processing

Stream Processing

Data Warehouse

Internet of Things

Scientific Data

Graph Database

Geographical Databases

Social Networks

Security

XML

Data Provenance
Today’s DBMS Hardware

Large main memory

Multi-core CPUs

Solid state disks

Co-processors
Future DBMSs

- Capacity of main memory < 1% of the stored data
  - **DB in main memory**
- Fixed block size based on the transfer unit
  - **Direct access of data on all devices**
- Central scheduler to schedule transactions
  - **Which processor should do the job?**
- No redundant data storage in main memory
  - **Redundant data at co-processors**
- Pipelining is always beneficial
  - **Co-processors like GPU support massive parallelism**
- Reuse of compiled plan
  - **Load-balancing between co-processors requires different plans**
The Goals of a "Databaser"

- Performance

Picture taken from [6]
The Goals of a "Databaser"

- Performance
- Performance

Picture taken from [6]
The Goals of a "Databaser"

- Performance
- Performance
- Performance

Picture taken from [6]
The Goals of a ”Databaser”

• Performance
• Performance
• Performance

How can we achieve more performance?

Picture taken from [6]
Trends for DBMS’s

- Use main memory as primary storage $\rightarrow$ Speed up data access

- Exploit all hardware capabilities such as co-processors $\rightarrow$ Speed up database operators
Are DBMSs written for yesterdays hardware efficient on todays hardware as well?
Are DBMSs written for yesterday's hardware efficient on today's hardware as well?

"30 years of Moore’s law has antiquated the disk-oriented relational architecture for OLTP applications" [?]
Data Access – Yesterday’s Bottleneck
Data Access – Today’s Bottleneck

Bottleneck

Bottleneck
The World of Co-Processors

PCI Express Bus

Picture taken from [7]
What do we have to change in DBMSs’ architecture to exploit new hardware capabilities and to meet tomorrow’s challenges?
Overview of Topics

1. Main-Memory DBMSs (Broneske, 2 lectures)
2. Advanced Query Optimization (Meister, 2 lectures)
3. Internet of Metaproteomics (Zoun, 1 lecture)
4. Main-Memory Index Structures (Broneske, 2 lectures)
5. Main-Memory Graph Algorithms (Pinnecke, 1 lecture)
6. Hardware-sensitive DB Algorithms (Gurumurthy, 2 lectures)
7. HTAP and Self-Driving Systems (Campero, 2 lectures)
Main-Memory DBMSs

Address main-memory access bottleneck using cache-conscious

- data layouts (e.g., column-stores [?]) and
- data processing [?]

Query processing in row-stores

```
<table>
<thead>
<tr>
<th>...</th>
<th>Location</th>
<th>...</th>
<th>Sales</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>MD</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EF</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MD</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EF</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

```
SELECT SUM(Sales)
FROM Shop
WHERE Location = 'MD'
```

Query processing in column-stores

```
Location
MD
EF
MD
EF
Year
2009
2010
2011
2010
Sales
1
5
7
4
```

```
σ_{Location = 'MD'}
```

```
σ_{Location = 'MD'}
```

```
Merge
```

```
SUM
```

```
SUM(Sales)
```

Adapted from [?]
Advanced Query Optimization

\[ r(\text{ORDER}) \bowtie r(\text{PRODUCT}) \bowtie r(\text{CUSTOMER}) \bowtie r(\text{SUPPLIER}) \]

Equivalent plans for one query \[\implies\] Large search space

- \((P \bowtie O) \bowtie S\)
- \((P \bowtie C) \bowtie S\)
- \((P \bowtie C) \bowtie O\)
- \(((P \bowtie S) \bowtie O) \bowtie S\)
- \(((P \bowtie C) \bowtie S) \bowtie O\)

Search space increases in modern systems
\[\implies\] Advanced optimization algorithms needed:
- Parallelization
- Usage of co-processors

Taken from [?]
Internet of Metaproteomics

- Biological field metaproteomics continuously produce data
  - Identification of spectra data is the required step of the workflow
- Increase the performance using Fast Data architecture
  - Mini Batch processing
  - Streaming messages
Main-Memory Index Structures

Access bottleneck → Search time in index structure matters

Optimizations:

- Compression
- Single Instruction Multiple Data (SIMD)
- Node specialization
- Explicit memory layout
Main-Memory Graph Algorithms

- Graph database fundamentals
- Operator pipeline: Imperative, declarative
- Graph traversals: Formal perspective
- Implementation strategies
- Physical operators for traversals
Hardware-sensitive DB Algorithms

CPU “smaller than”-selection

```c
int pos = 0;
for (int i = 0; i < array.size; ++i) {
    if (array[i] < comp_val) {
        result[pos++] = i;
    }
}
```

GPU “smaller than”-selection

```c
int tid = threadIdx.x + blockIdx.x * blockDim.x;
while (tid < array.size) {
    bitmask[tid] = (array[tid] < comparison.value);
    tid += blockDim.x * gridDim.x;
}
```

Optimized Code differs between processing devices w.r.t.

- Access pattern
- Code optimizations (e.g., branch-free code)
- Parallelization capabilities
Analytical (OLAP) and Transactional processing (OLTP):
• Different access patterns
• Different performance goals (i.e., latency vs. throughput)
⇒ Different optimizations required

Hybrid Transactional/Analytical Processing (HTAP):
⇒ a challenge for DBMSs
Self-Managing Solutions to the HTAP Challenge

- Advanced storage engine designs exist for HTAP\(^1\).
- Self-managing features are important in these designs, enabling the DBMS to adapt to changing workloads.

---

\(^1\) For examples, see: [?], [?], [?]
Organization

- Lecture: Friday, 13:15 - 14:45, Raum G10-460
  - Lecturer: David Broneske, Gabriel Campero, Bala Gurumurthy, Andreas Meister, Marcus Pinnecke, Gunter Saake, Roman Zoun
  - Each lecturer will provide insights about his own research topics
    - Office hours: upon consultation
    - Email: firstname.lastname@ovgu.de

- Exercise: Wednesday, 15:15 - 16:45, Raum G10-111
  - Starting 18.04.2018
  - Tutor: Andreas Meister
    - Office hours: upon consultation
    - Room: G29-108
    - Email: andreas.meister@ovgu.de
Organization

- Information (time, rooms) & slides available on www.dbse.ovgu.de/Lehre/ATDB.html
elearning.ovgu.de/course/view.php?id=4061

- Examination
  - Likely written examinations
  - 120 minutes
  - Questions about lecture and exercise
  - Requirements:
    - Registration for exercises via LSF until 29.04.2017
    - Assigned minimal **60%** of all exercise tasks
    - Presentation of at least **3** tasks in the exercise
Exercise

• Task based on lecture content
• Available at E-Learning
• Requirements:
  • Preparation
  • Understanding
  • Presentations
• Plagiarism not allowed → Tasks will not be accepted
References I

Column-Stores vs. Row-Stores: How different are they really?
In SIGMOD, pages 967–980.

Bridging the archipelago between row-stores and column-stores for hybrid workloads.
In Proceedings of the 2016 International Conference on Management of Data, pages 583–598. ACM.

Towards a one size fits all database architecture.
In CIDR, pages 195–198.

Hyper: A hybrid oltp&olap main memory database system based on virtual memory snapshots.
In Data Engineering (ICDE), 2011 IEEE 27th International Conference on, pages 195–206. IEEE.

Data Warehouse Technologien.
mitp-Verlag.

References II


Web Resources


