Database Systems and Information Management Group
Fak. IV Electrical Engineering and Computer Science
Technische Universität Berlin

Bachelor's and Master's Thesis Topics

Academic Year 2021/22

Last Updated: April 26, 2022

Table of Contents

Foreword

This document contains a compilation of thesis topics for both Bachelor's and Master's students pursuing a degree at the Technische Universität Berlin. It reflects the current research activities in the Chair of Database Systems and Information Management (DIMA). Students should identify a topic or two and contact the respective advisor via email. Advisor email addresses are listed on this webpage: https://www.dima.tu-berlin.de/menue/team/.

Many DIMA advisors are already working with several students and it may be that they do not currently have the capacity to take on another student. For this reason, you will need to inquire about their availability. In several cases, the topics listed in this document can be customized based on your knowledge and skills. To ease the initial meeting, please complete and forward the *Thesis Request Form*[1] (Table 1) to your prospective advisor. If you have any questions feel free to reach out to Juan Soto (juan.soto@tu-berlin.de) for an appointment.

We wish you great success!

Juan Soto
Academic Director, DIMA

Student Name (Surname, First Name)		Mustermann, Erika
Thesis Level	• B.Sc. or M.Sc.	• M.Sc.
Thesis Topic	• research area(s)	• databases for emerging hardware
Technical Skills	• programming languages	• C++, Java, Scala, Python, R
	• systems	• Flink, Hadoop, Spark
	• software	• DB2, Oracle, Postgres
Foundational Knowledge	• B.Sc. courses completed (marks)	• ISDA (2,0), DBPRA (1,3), DBPRO (1,7), • DBSEM (1,3), DW (1,0)
	• M.Sc. courses completed (marks)	• DBT (1,7), IDBPRA (2,0), AIM-2 (1,3), AIM-3 (1,7), • BDAPRO (1,3), BDASEM (1,3), MHD (1,0), ROC (1,3)
Practical Experience (only complete those that are appropriate)	• research assistant	• Conducted research in query optimization at ...
	• open-source code contributions	• Code contributions to **Apache ProjectX**, https://github.com/Mustermann_Erika/ProjectX/
	• student intern	• Interned at Google in summer 2019, worked on ...
	• full-time employment	• As a BI Analyst (Deutsche Telekom), I performed ...

Table 1. A representative example of a completed Thesis Request Form.

[1] The *Thesis Request Form* as well as the DIMA *Thesis Proposal Template* are both available for download from this webpage: https://www.dima.tu-berlin.de/menue/thesis_opportunities/.

Thesis Opportunities in the DIMA Group

Theses in the DIMA Group are *often tied to ongoing research projects* sponsored by (inter-) national funding agencies and are *commonly written in English (and some in German).* Problems are typically centered on topics in database systems as well as scalable and distributed data management, including:

- benchmarking and performance evaluation,
- data visualization,
- data warehousing, OLAP, SQL Analytics,
- database monitoring and tuning,
- database security, privacy, access control,
- databases for emerging hardware,
- data systems and data management for machine learning,
- distributed and parallel databases,
- graph data management, RDF, social networks,
- knowledge discovery, clustering, data mining,
- machine learning for data management and data systems,
- query processing and optimization,
- spatio-temporal databases,
- storage, indexing, and physical database design,
- streams, sensor networks, complex event processing,
- transaction processing,
- very large data science applications/pipelines.

To pursue a thesis with us, students are generally required to possess:

- *outstanding programming skills* in C++, Java, or Scala,
- *extensive knowledge in database systems* (e.g., IBM DB2, Oracle) or *big data analytics systems* (e.g., Flink, Spark),
- *basic knowledge in the use of an IDE* (e.g., Eclipse, IntelliJ),
- *basic knowledge in the use of a distributed version control system* (e.g., SVN, Git).

Furthermore, to conduct a:

- **Bachelor's thesis**, students *must have successfully completed* **ISDA** and **DBPRA** (at a minimum) *with a grade of good or better* and possibly several other Bachelor's courses offered by DIMA, such as DBPRO, DBSEM, or DW.
- **Master's thesis**, students *must have successfully completed* **DBT** and **IDBPRA** (at a minimum) *with a grade of good or better* and possibly several other Master's courses offered by DIMA, such as AIM-2, AIM-3, BDAPRO, BDASEM, MHD, or ROC.

Moreover, depending on the thesis topic, additional knowledge may be required (e.g., compiler technology, distributed systems, machine learning).

1. Compliant Data Processing

Advisor: Dr. Kaustubh Beedkar

Appropriate Target Level: B.Sc., M.Sc.

Keywords: geo-distributed data analysis, compliant query processing, federated databases, data provenance

Description: Many large organizations today operate data centers that produce large amounts of data at different locations around the globe. Analyzing geographically distributed data is essential to derive valuable insights. Typically, geo-distributed data analysis is carried out by either communicating all of the data to a central location, where analytics are performed or employing a distributed execution strategy to minimize data communication. However, legal constraints arising from regulatory bodies concerning data sovereignty and data movement (e.g., prohibit the transfer of data across national borders) as well as technical constraints arising from the use of heterogeneous compute nodes pose serious limitations to existing approaches. Our research explores how to declaratively specify the algorithms as well as the legal and technical constraints to automatically derive distributed execution strategies. We are also interested in exploring methods, such as those based on data provenance to audit compliance with data policies in decentralized execution environments.

Prerequisites: *strong programming skills* (preferably in Java); *knowledge in query planning and execution in DBMS*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

2. Data Stream Modeling and Processing

Advisor: Dr. Alexander Borusan

Appropriate Target Level: B.Sc., M.Sc.

Keywords: data stream management in embedded information systems

Description: Automotive, avionic, and manufacturing applications built atop embedded information systems, typically include two tasks: *monitoring* and *controlling*. Many of these applications continuously generate data streams, which are ordered sequences of data that can only be read once and ideally processed in real-time. From the point of view of data stream management, several tasks need to be solved, in order to: *model the data, process the data* (e.g., incorporating appropriate data structures, employing data reduction techniques), *querying the data*, *scheduling jobs*, as well as *storing the data*. Additionally, over the past decade, *data stream analysis* has become one of the most important tasks.

Examples of topics that could be explored, include

- M.Sc. Thesis: An examination of existing and the development of novel architectures and models for the real-time processing of streaming data originating in embedded information systems. For example, for systems that are employed in the automotive or avionics sector.

- B.Sc. Thesis: An analysis of data reduction techniques suitable for automotive applications involving data streams, including the development of a taxonomy.

Prerequisites: *strong programming skills in Java*; *good writing skills*; *knowledge in Apache Flink*; *knowledge in data streams management systems* (DSMS); (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA); *solid mathematics background*.

3. Data Analytics for Massive Time Series

Advisor: Dr. Holmer Hemsen

Appropriate Target Level: B.Sc., M.Sc.

Keywords: scalable signal processing

Description: A time series is a set of observations each recorded at a specific time. Examples of time series are many fold (e.g., electrocardiography curves, stock market, seismic measurements, network load). Time series analysis comprises a wide range of methods, such as *anomaly and outlier detection*, *forecasting*, and *pattern recognition*. The focus of this topic area is on research of methods for the analysis of massive and/or multi-dimensional time series.

Prerequisites: *strong programming skills* in Java, Scala or Python; *good writing skills*; (preferably) *knowledge in Apache Flink*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

4. Intelligent and Scalable Resource Management for Industrie 4.0

Advisor: Dr. Holmer Hemsen

Appropriate Target Level: B.Sc., M.Sc.

Keywords: Industrie 4.0

Description: The goal of Industrie 4.0 is to digitalize, automate and optimize industrial production systems. In many cases, this involves upgrading conventional production systems into cyber-physical systems, often drawing on Internet of Things (IoT) technologies. The focus of this research topic is on methods for the scalable optimization of production lines and the intelligent forecasting of consumable resources to calculate optimal dynamic maintenance strategies.

Prerequisites: *strong programming skills* in Java, Scala or Python; *good writing skills*; (preferably) *knowledge in Apache Flink*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

5. Using Machine Learning for Query Optimization

Advisor: Dr. Zoi Kaoudi

Appropriate Target Level: B.Sc., M.Sc.

Keywords: machine learning for data management

Description: With the recent successes attributed to the use of machine learning (ML) in varying domains, there is an increased interest in employing ML for query optimization (e.g., cardinality estimation [1, 2], cost estimation [3, 4]). In this thesis, we seek to develop and evaluate ML-based techniques for query optimization that outperform query optimizers in *commercial DBMS*, such as SQL server and IBM DB2, as well as in *big data systems*.

Prerequisites: *strong programming skills* (e.g., Java, SQL); *good writing skills*; (preferably) *knowledge in big data systems* (e.g., Apache Flink, Apache Spark) *and machine learning*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Parimarjan Negi, Ryan C. Marcus, Andreas Kipf, Hongzi Mao, Nesime Tatbul, Tim Kraska, Mohammad Alizadeh: Flow-Loss: Learning Cardinality Estimates That Matter. Proc. VLDB Endow. 14(11): 2019-2032 (2021)

[2] Jie Liu, Wenqian Dong, Dong Li, Qingqing Zhou: Fauce: Fast and Accurate Deep Ensembles with Uncertainty for Cardinality Estimation. Proc. VLDB Endow. 14(11): 1950-1963 (2021)

[3] Ryan C. Marcus, Parimarjan Negi, Hongzi Mao, Chi Zhang, Mohammad Alizadeh, Tim Kraska, Olga Papaemmanouil, Nesime Tatbul: Neo: A Learned Query Optimizer. Proc. VLDB Endow. 12(11): 1705-1718 (2019)

[4] Zoi Kaoudi, Jorge-Arnulfo Quiané-Ruiz, Bertty Contreras-Rojas, Rodrigo Pardo-Meza, Anis Troudi, Sanjay Chawla: ML-based Cross-Platform Query Optimization. ICDE 2020: 1489-1500

6. Combining Machine Learning with Reasoning

Advisor: Dr. Zoi Kaoudi

Appropriate Target Level: B.Sc., M.Sc.

Keywords: knowledge graph embeddings, logical reasoning

Description: Knowledge graphs are extensively used in many application domains, such as search engines, product recommendation, and bioinformatics. Knowledge graph completion (a.k.a. link prediction) refers to the task of inferring missing information, such as links. To achieve link prediction, *knowledge graph embeddings* (KGE) [1,2] are gaining popularity. However, KGE are data-driven and do not perform well for sparse entities [3]. An alternative is to employ *rule-based approaches*, which learn rules from a knowledge graph (e.g., [4]), and then applies them to infer the missing links. However, this approach is also data-driven and one cannot learn rules when there is insufficient evidence. Hence, to overcome these problems, we seek to develop a novel approach that combines *knowledge graph embeddings* with *rule-based* reasoning, where the rules are based on ontologies, determined by domain experts, or specified by entailment regimes (e.g., RDFS, OWL2).

Prerequisites: *strong programming skills* (e.g., Python, Java); *good writing skills*; (preferably) *knowledge of embeddings and first order reasoning*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, Oksana Yakhnenko: Translating Embeddings for Modeling Multi-relational Data. NIPS 2013: 2787-2795

[2] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, Guillaume Bouchard: Complex Embeddings for Simple Link Prediction. ICML 2016.

[3] Aisha Mohamed, Shameem Puthiya Parambath, Zoi Kaoudi, Ashraf Aboulnaga: Popularity Agnostic Evaluation of Knowledge Graph Embeddings. UAI 2020: 1059-1068

[4] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. WWW 2013: 413–422

7. Scalable Machine Learning Systems for Streaming Graphs

Advisor: Dr. Zoi Kaoudi

Appropriate Target Level: B.Sc., M.Sc.

Keywords: data management for machine learning systems

Description: Many real-world applications require machine learning on graphs that are continuously changing (i.e., *streaming graphs*). Examples include *social networks* (where new friendships are perpetually created), *applications that monitor and predict new connections* (to improve recommendations), and the *Internet of Things* (where millions of participants change their physical location continously). In each of these cases, we need solutions that are able to cope with the highly-dynamic behavior that is prevalent. We seek to develop a system that can run analytics and machine learning tasks over streaming graphs.

Prerequisites: *strong programming skills* (e.g., Java); *good writing skills*; (preferably) knowledge in Apache Flink; (ideally) *completed several DIMA courses* (e.g., AIM-3, BDAPRO, BDASEM, DBT, DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA); *machine learning knowledge*.

8. Intelligent Query Dissemination in the IoT

Advisor: Dr. Varun Pandey

Appropriate Target Level: M.Sc.

Keywords: IoT stream processing, indexes, dynamic nodes, spatial processing

Description: Cloud computing has been at the center stage over the past decade mainly due to offering centralized large-scale computing capabilities to lower the capital as well as computing costs for organizations. Large-scale cloud vendors have a few very large data centers spread across the globe. Cloud computing naturally led to various successful services being invented, deployed, and tailored for the cloud. Cloud-based stream processing systems are one of them, but are rather centralized (i.e., they require the data to be in the cloud so as to be processed). In the meantime, emerging applications for the Internet of Things (IoT), such as autonomous vehicles, mobile computing, and mobile sensor networks are the driving force behind dispersed computing or edge computing (the data is processed closer to the source of data). We are building NebulaStream (NES) [1], a stream processing platform capable of processing the data at the edge. In the context of NES, we require the dissemination of intelligent queries to the relevant (sensor) nodes. Regarding this, there are two concrete thesis topics available: (i) *spatial indexes for frequent updates*, and (ii) *lightweight indexes for query dissemination in IoT*.

Expected Outcomes:

- Investigate the performance of the indexes in the context of stream processing

- Improve the performance of indexes using well-know techniques

- Evaluate the implementation and conduct a performance analysis

Prerequisites: *good programming skills* in Java or C++; *good writing skills*; (ideally) *some experience in big data frameworks*; (ideally) *completed several DIMA courses* e.g., AIM-3, BDAPRO, BDASEM, DBT, DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Zeuch, Steffen, et al. "The NebulaStream Platform: Data and Application Management for the Internet of Things." Conference on Innovative Data Systems Research (CIDR 2020), January 2020, https://www.nebula.stream/publications/nebulastreamCIDR.html

9. Machine Learning for Databases

Advisor: Dr. Varun Pandey

Appropriate Target Level: M.Sc.

Keywords: database processing, machine learning, ML for DB

Description: Triggered by advances in processing power, memory, storage, and computing technologies, machine learning has been at the forefront of many research areas. This has also led to machine learning making its way into database research. Tuning databases for optimal performance is a cumbersome task, and requires either human intervention (DBA's) or empirical optimization techniques (e.g., index selection, cardinality estimation). Machine learning techniques are adept in learning various patterns in data, and are thus suited for many such internal optimizations in databases. One particular application of machine learning in database research is *indexing multi-dimensional data*. Recent work on multi-dimensional indexing has focused on partitioning the space [2][3], and sorting the values inside each partition on the sort dimension, and learning a regression model on the sorted values. The sort dimension is tailored to a particular workload, and thus if the workload changes (more queries on some other dimension/attribute), it requires an expensive re-partitioning step and retraining the models for each partition. In this thesis, we will investigate ways to avoid the expensive re-partitioning step.

Note: Students are also encouraged to define their own thesis agenda. Although my main interest lies in multi-dimensional indexing, I am open to other areas of application of machine learning in databases. Students are encouraged to read [1], and investigate the areas where machine learning has been applied in databases (i.e., see section AI for DB in [1]).

Prerequisites: *good programming skills* in Java or C++; *good writing skills*; (ideally) *some experience in big data frameworks*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA); *machine learning knowledge*.

References

[1] Zhou, Xuanhe, et al. "Database meets artificial intelligence: A survey." IEEE Transactions on Knowledge and Data Engineering (2020).

[2] Nathan, Vikram, et al. "Learning multi-dimensional indexes." Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. 2020.

[3] Ding, Jialin, et al. "Tsunami: a learned multi-dimensional index for correlated data and skewed workloads." Proceedings of the VLDB Endowment 14.2 (2020): 74-86.

10. Towards Optimal Physical Layout for Efficient Query Processing in the Cloud

Advisor: Dr. Varun Pandey

Appropriate Target Level: M.Sc.

Keywords: cloud computing, machine learning, ML for DB

Description: Businesses today seek to run analytics on large amounts of captured data efficiently, to gain insights into their business practices. Increasingly, these analytics have been run on cloud based data warehouse systems. The essential technology that these cloud warehouses leverage is the separation of compute from storage. Network based object storage, such as S3, is preferred by these warehouses for cheap storage. To run queries, these warehouses can spin up compute instances on-demand, based on user requirements and query the data from S3. For efficient query processing, it is necessary that a minimum amount of data (objects or files in context of S3) is transferred from S3 to minimize expensive data transfer over the network. To this end, cloud warehouses employ data skipping techniques, such as range based small materialized aggregates (SMA) or Zone-maps [1], or by ordering the data on some important attribute (such as the timestamp). Although multidimensional indexes have been a very popular research area over the past four decades, they have not made their way into mainstream cloud warehouses, such as Google Big Query and Snowflake. Slowly these multidimensional indexes have made their way into these services, such as Amazon Redshift [2], Apache Spark [3], and Oracle [4]. Some of these services utilize Z-ordering to cluster user data, but require users to specify the order. In this thesis, we aim to empirically quantify why these multi-dimensional indexes could be better or worse than a range-based index in one dimension.

Prerequisites: *good programming skills* in Java or C++; *good writing skills*; (ideally) *some experience in big data frameworks*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] G. Moerkotte. Small Materialized Aggregates: A light weight index structure for data warehousing. In VLDB, pages 476–487, 1998.

[2] Amazon AWS. 2016. Amazon Redshift Engineering's Advanced Table Design Playbook: Compound and Interleaved Sort Keys. https://aws.amazon.com/blogs/big-data/amazon-redshiftengineerings-advanced-table-design-playbook-compound-andinterleaved-sort-keys/.

[3] Databricks Engineering Blog. Processing Petabytes of Data in Seconds with Databricks Delta. https://databricks.com/blog/2018/07/31/processing-petabytes-of-data-in-seconds-with-databricks-delta.html.

[4] Ziauddin, Mohamed, et al. "Dimensions based data clustering and zone maps." Proceedings of the VLDB Endowment 10.12 (2017): 1622-1633.

11. Big Data Processing

Advisor: Dr. Jorge Quiane Ruiz

Appropriate Target Level: B.Sc., M.Sc.

Keywords: scalable data management

Description: Increasingly, applications need to selectively extract relevant data of interest from large volumes of diverse datasets generated at high velocity. In order to cope with today's applications needs and meet the demand, scalable data processing tools and techniques must be employed.

If you have ever wondered how:

(a) *databases cope with complex data*?
(b) *dataflow systems* (e.g., Flink or Spark) *work*?
(c) *to effectively use existing big data systems*?
(d) *big data can help a company or organization*?
(e) *big data helps machine learning*?
(f) *big data will impact the future*?

…. then this area of research will be of interest to you.

Specific Thesis Topics:

1. Benchmarking Intermediate Data Representation for Fast Data Transfers
2. Scalable Trusted Data Processing

Prerequisites: *strong programming skills in Java*; *good writing skills*; *knowledge in Apache Flink*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

12. Data-Related Ecosystem

Advisor: Dr. Jorge Quiane Ruiz

Appropriate Target Level: B.Sc., M.Sc.

Keywords: scalable data management

Description: Today, data intelligence is monopolized by just a few companies, given that they possess both *sizable amounts of data* and *technological solutions*, to address big data challenges and solve artificial intelligence (AI) problems. Therefore, it is vital to provide novel ways to share data and leverage big data/AI technologies, such as an ecosystem, so that everyone can benefit from the data intelligence era. However, building such an ecosystem is quite challenging, since we do not yet have the right data infrastructure. If this appeals to you, join the Agora Project, which aims to devise a suitable data infrastructure and make a data-related ecosystem possible.

Specific Thesis Topics:

1. Benchmarking Monitoring Systems for Widely-Open Distributed Environments
2. Visual Data Processing

Prerequisites: *strong programming skills in Java*; *good writing skills*; *knowledge in Apache Flink*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

13. Data Debugging

Advisor: Dr. Jorge Quiane Ruiz

Appropriate Target Level: M.Sc.

Keywords: scalable data management

Description: How many times have you heard that *big data* (bd), *data science* (ds), or *machine learning* (ml) are helping scientists (from varying domains) to make great progress? But, have you ever heard (even once) how hard it is to get those bd, ds, or ml pipelines shiny (i.e., readily accessible) for the applied practitioner? The reality is debugging bd, ds, or ml pipelines is 'taboo': nobody wants to talk about it, but we all suffer from it! Data debugging seeks to break this taboo. We are developing a general-purpose data debugging system that enables the interactive debugging of bd, ds, and ml pipelines.

Specific Thesis Topics:

1. Interactive Data Debugging
2. Visual Data Debugging

Prerequisites: *strong programming skills in Java*; *good writing skills*; *knowledge in Apache Flink*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

14. An Analysis of Data Analytics Libraries

Advisor: Juan Soto

Appropriate Target Level: B.Sc., M.Sc.

Keywords: data analytics, machine learning, quality assurance, mathematical software

Description: The most recent *Data and AI Landscape* [1] illustrates the growing number of data analytics (e.g., machine learning) libraries [2,3,4,5,6,7,8] that are increasingly available. These libraries are commonly employed in data science pipelines and comprised of numerous underlying mathematical and statistical libraries. However, more research needs to be undertaken to *assess their correctness* (in light of diverse datasets) or *guarantee that they are of high-quality* (e.g., able to cope with machine precision or roundoff errors, employ reliable underlying mathematical software). In this thesis, we want to take a closer look at the reliability of data analytics libraries. The approach to be undertaken includes conducting a theoretical study of the selected algorithms, the generation of synthetic data and selection of appropriate datasets, as well as performing empirical/numerical studies to assess quality.

Prerequisites: good programming skills; *software testing*; *familiarity with mathematical, statistical, or data analytics libraries*; (ideally) *completed course work in algorithms, (numerical) linear algebra, machine learning, numerical analysis, probability and statistics as well as DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Data and AI Landscape 2020. https://shorturl.at/cghvX.
[2] Intel® oneAPI Data Analytics Library. https://shorturl.at/kGMZ9.
[3] Apache MADlib: Big Data Machine Learning in SQL. https://madlib.apache.org/.
[4] Apache Mahout. https://mahout.apache.org/.
[5] H_2O Machine Learning Platform, https://www.h2o.ai/
[6] ScalaNLP Suite of Libraries. http://www.scalanlp.org/.
[7] Apache Spark MLlib. https://spark.apache.org/mllib/.
[8] Apple Turi Create. https://github.com/apple/turicreate.
[9] SE4ML – Software Engineering for AI-ML-based Systems, Dagstuhl Seminar 20091.
[10] Machine Learning Testing: Survey, Landscapes and Horizons. https://shorturl.at/qtyK5a and https://doi.org/10.1109/TSE.2019.2962027.
[11] Shin Nakajima. Quality Assurance of Machine Learning Software. GCCE 2018.
[12] Tutorial on Software Testing & Quality Assurance for Machine Learning Applications from research bench to real world. CoDS COMAD 2020. https://shorturl.at/bfkBP (Short Paper) and https://shorturl.at/dBV25 (Tutorial Materials).
[13] Numerical Issues in Statistical Computing for the Social Scientist. December 2003. https://shorturl.at/irCRW.

15. Query Optimization, Processing and Execution on Modern CPUs

Advisor: Dr. Steffen Zeuch

Appropriate Target Level: M.Sc.

Keywords: query optimization, query execution on modern CPUs

Description: Over the past decades, database systems have migrated from *disk* to *memory architectures*, such as RAM, Flash, or NVRAM. Research has shown that this migration fundamentally shifts the performance bottleneck upwards in the memory hierarchy. Whereas *disk-based* database systems were largely dominated by disk bandwidth and latency, *in-memory* database systems mainly depend on the efficiency of faster memory components (e.g., RAM, caches, registers). With respect to hardware, the 'clock speed per core' reached a plateau due to physical limitations. To overcome this limit, hardware architects dedicated an increasing number of available on-chip transistors to processors and caches. Unfortunately, the improvements to the *memory bandwidth* far outpaced the improvements to the *memory access latency*. Nowadays, CPUs are able to process data far faster than they are able to transfer data from main memory to caches. Consequently, this trend is referred to as the *memory wall*, which is the main challenge in modern main memory database systems. In this proposed thesis, students will seek to reduce the impact of the memory wall and realize the full potential of the available processing power in modern CPUs for database systems.

Note: Students are encouraged to propose their own topic in the field of query optimization, processing, and execution on modern CPUs.

Prerequisites: *strong programming skills* in C/C++; *good writing skills*; *deep knowledge in database implementation techniques*; *good understanding of computer architecture*; (ideally) knowledge in LLVM, Vtune, MPI, and OpenMP; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DMH (formerly MHD), DBPRA, DBPRO, DBSEM, ISDA).

16. Machine Learning Approaches in Non-Intrusive Load Monitoring

Advisor: Xenofon Chatziliadis

Appropriate Target Level: M.Sc.

Keywords: data stream processing, load monitoring, machine learning, deep learning

Description: In a common monitoring use case for stream processing engines (SPEs) various system metrics (e.g. CPU utilization, memory consumption, network traffic) and application metrics (e.g., tuple throughput) must be transmitted from workers of the topology to the corresponding coordinator node. In a large-scale scenario, like the IoT (Internet of Things) it becomes, however, a major bottleneck when several thousands of nodes intend to transmit monitoring data simultaneously to a single destination. One possible approach to reduce the amount of data is aggregation. In order to maintain the accuracy of the aggregated data the worker nodes pre-aggregate different kinds of monitoring metrics to a single-channel time series. The aggregated data is then transmitted to the coordinator node, where it is then decomposed into the original metrics. In this thesis, students will need to analyze the applicability of state-of-the-art nonintrusive load monitoring (NILM) techniques firstly introduced by Hart [3]. The main idea of NILM is to decompose single-channel aggregated time series into the individual appliances, i.e., this corresponds to the disaggregation process that happens at the coordinator node (see Fig. 1). The state-of-the-art NILM disaggregation techniques are based on different machine learning (ML) techniques (e.g., Denoising Autoencoders [5], Hidden Markov Models [6], Convolutional Neural Networks [7], Long Short-Term Memory / LSTM Neural Networks [8]).

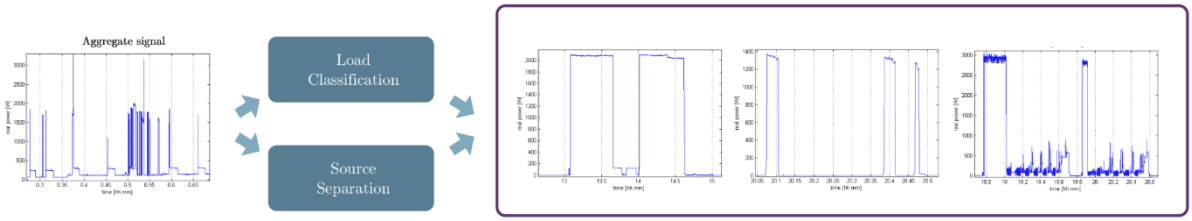


Fig. 1: Load disaggregation using NILM where a single-channel time series is decomposed into three appliances. [4]

NILM has been successfully applied in various domains of the energy sector (e.g., smart meters). Its applicability however for internal data reduction in SPEs has not been tested so far. The goal is to test existing NILM techniques in the area of performance and application monitoring for SPEs. The requirements of this thesis are more precisely as follows:

1. Give an overview of state-of-the-art NILM techniques and identify the best suitable approach for given stream processing workloads (e.g., Yahoo Streaming Benchmark).
2. Create monitoring data of the specified workloads on a NebulaStream (NES) topology, which will be used as training data for the NILM ML models.
3. Create features that can be used to train a NILM model based on 2.
4. Evaluate the performance of the NILM model based on elaborated criteria, such as accuracy, training time, and required processing resources.

Prerequisites: *good programming skills* (e.g., Python, Java, Scala, C++); *experience with distributed management in large-scale systems* is highly desired; *advanced knowledge in DNNs and ML* is mandatory; *good writing skills*; (ideally) *completed coursework* at DIMA (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA) or possess the equivalent knowledge in distributed systems, stream processing engines, and databases as well as *completed coursework in machine learning*.

References

[1] NebulaStream: Complex Analytics Beyond the Cloud, https://www.nebula.stream/paper/zeuch_vliot.pdf

[2] Monitoring of Stream Processing Engines Beyond the Cloud: An Overview, https://www.nebula.stream/paper/chatziliadis_vliot2021.pdf

[3] G.W. Hart. Nonintrusive appliance load monitoring. Proceedings of the IEEE, 80(12):1870–1891,1992.

[4] Unsupervised Algorithms for Non-Intrusive Load Monitoring: an Up-to-Date Overview, https://www.researchgate.net/profile/Stefano-Squartini/publication/277009814_Unsupervised_Algorithms_for_Non-Intrusive_Load_Monitoring_an_Up-to-Date_Overview/links/55c4eb8208aeca747d6182c5/Unsupervised-Algorithms-for-Non-Intrusive-Load-Monitoring-an-Up-to-Date-Overview.pdf

[5] Oliver Parson, S Ghosh, M Weal, and A Rogers. Non-intrusive Load Monitoring using Prior Models of General Appliance Types. In Proceeding of the twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI-12), Toronto, Canada, Apr 2012.

[6] H. Kim, M. Marwah, M. F. Arlitt, G. Lyon, and J. Han. Unsupervised Disaggregation of Low Frequency Power Measurements. Proceedings of the 11th SIAM International Conference on Data Mining, pages 747–758, 2011.

[7] Y. Jia, H. Wang, N. Batra, and K. Whitehouse, “A tree-structured neural network model for household energy breakdown,” Web Conf. 2019 - Proc. World Wide Web Conf. WWW 2019, pp. 2872–2878, 2019.

[8] J. Kelly and W. Knottenbelt, “Neural NILM: Deep Neural Networks Applied to Energy Disaggregation,” 2015.

17. On-demand Gathering in Bandwidth Constrained IoT Networks

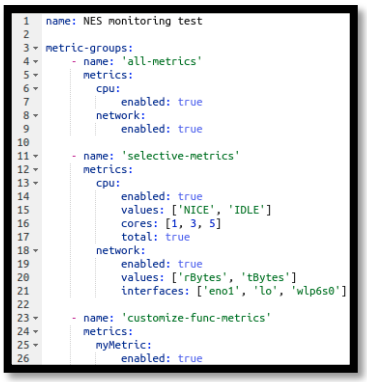
Advisor: Xenofon Chatziliadis

Appropriate Target Level: B.Sc., M.Sc.

Keywords: data stream processing, adaptive load monitoring, on-demand data retrieval

Description: IoT networks consist of battery-powered and resource constrained devices that are required to operate for long periods of time. Reducing energy drain is of the utmost importance when performing computational operations on such edge devices. By avoiding the transmission of unnecessary data packets, the lifetime of battery-powered devices can be increased while additionally decreasing the utilization of bandwidth in the network [1, 2, 3]. In this thesis, students will need to test existing demand-based retrieval approaches in the context of performance and application monitoring in NebulaStream (NES) [4]. In particular:

1. Give an overview of the state-of-the-art techniques that reduce bandwidth utilization in sensor networks like in [1, 2, 3].
2. Create monitoring data of specified workloads (e.g., Yahoo Streaming Benchmark) on an NES topology. Alternatively, publicly available monitoring data from Google's data centers can be used [5].
3. Extend the monitoring API of NES [6] to enable configurable gathering of performance metrics via personalized configurations (e.g., YAML).
4. Evaluate the performance of the implemented bandwidth reduction techniques by measuring and comparing bandwidth and computational load of the topology between the different approaches.



Prerequisites: *good programming skills* (e.g., Python, Java, Scala, preferably C++); *experience with stream processing and distributed management in large-scale systems* is highly desired; *good writing skills*; (ideally) *completed coursework* at DIMA (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Samir Goel and Tomasz Imielinski. 2001. Prediction-based monitoring in sensor networks: taking lessons from MPEG. SIGCOMM Comput. Commun. Rev. 31, 5 (October 2001), 82–98. DOI: https://doi.org/10.1145/1037107.1037117

[2] Antonios Deligiannakis, Yannis Kotidis, and Nick Roussopoulos. 2008. Bandwidth-constrained queries in sensor networks. The VLDB Journal 17, 3 (May 2008), 443–467. DOI: https://doi.org/10.1007/s00778-006-0016-z

[3] https://link.springer.com/content/pdf/10.1007/s00778-004-0138-0.pdf

[4] NebulaStream: Complex Analytics Beyond the Cloud, https://www.nebula.stream/paper/zeuch_vliot.pdf

[5] More Google Cluster Data, https://ai.googleblog.com/2011/11/more-google-cluster-data.html

[6] Monitoring of Stream Processing Engines Beyond the Cloud: An Overview, https://www.nebula.stream/paper/chatziliadis_vliot2021.pdf

18. Synopses and Approximate Monitoring for Stream Processing

Advisor: Xenofon Chatziliadis

Appropriate Target Level: B.Sc., M.Sc.

Keywords: data stream processing, approximate load monitoring, synopses in data streams

Description: Processing data streams in large-scale IoT networks requires tight data handling and data dissemination techniques. Transmitting a full-resolution data stream from each source to a sink might be severely limited due to: (1) limited bandwidth that may be insufficient to support a continuous stream and (2) increased power consumption due to the multi-hop communication. A well-known idea to minimize the volume of transmitted data while maintaining historical information is *approximation*. Here, the data stream is replaced by an approximate signal tailored to the application. The trade-off is then between the size of the approximate signal and its precision compared to the real data [1]. In this thesis, students will need to test existing approximation techniques for time series, like Wavelets, Histograms and the Discrete Cosine Transform for their applicability for monitoring a stream processing engine, like NebulaStream (NES) [3]. In particular:

1. Give an overview of the state-of-the-art techniques that reduce bandwidth utilization in stream processing scenarios through approximation like in [1, 2]. (Note: The hierarchical structure of IoT networks should be considered when reviewing literature. Additionally, the algorithms must be able to approximate historical data.)
2. Implement the most promising approaches.
3. Evaluate the performance of the implemented approaches in terms of precision, memory utilization, computational overhead, and throughput.
4. For evaluation the Google monitoring dataset can be used [4].

Prerequisites: *good programming skills* (e.g., Python, Java, Scala, preferably C++); *experience with stream processing and distributed management in large-scale systems* is highly desired; *basic knowledge of statistics*; *good writing skills*; (ideally) *completed coursework* at DIMA (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Antonios Deligiannakis, Yannis Kotidis, and Nick Roussopoulos. 2007. Dissemination of compressed historical information in sensor networks. The VLDB Journal 16, 4 (October 2007), 439–461. DOI: https://doi.org/10.1007/s00778-005-0173-5

[2] A Survey of Synopsis Construction in Data Streams, http://www.charuaggarwal.net/synopsis.pdf

[3] NebulaStream: Complex Analytics Beyond the Cloud, https://www.nebula.stream/paper/zeuch_vliot.pdf

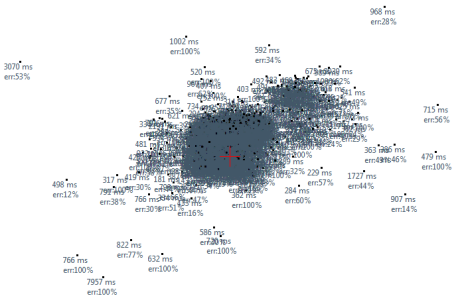
[4] More Google Cluster Data, https://ai.googleblog.com/2011/11/more-google-cluster-data.html

19. Distributed Network Embeddings for Large-scale IoT Topologies

Advisor: Xenofon Chatziliadis

Appropriate Target Level: B.Sc., M.Sc.

Keywords: data stream processing, network coordinate systems, distributed systems, ML



Description: Large-scale stream processing engines, like NebulaStream (NES) [1] require several monitoring metrics in order to be efficient [2]. A key challenge is monitoring round-trip times between devices. Such metrics can be predicted accurately and efficiently by network coordinate systems (NCS) [3]. NCS assign synthetic coordinates to hosts such that the distance between the coordinates of two hosts accurately predict the communication latency between the hosts. Well-known distributed algorithms for NCS include Vivaldi [4], PIC [5], NPS [6] and WDCS [7]. In this thesis, students will test existing distributed algorithms for NCS and evaluate their performance in NES [4]. In particular:

1. Give an overview of state-of-the-art approaches for NCS.
2. Extend the monitoring API of NES [2] to enable the prediction of network metrics via one of the mentioned approaches preferably Vivaldi [4].
3. Evaluate the performance of the implemented approach by measuring and comparing bandwidth, computational load, and accuracy in terms of scalability and convergence time.

Prerequisites: *good programming skills* (e.g., preferably C++); *experience with stream processing and distributed management in large-scale systems* is highly desired; *basic knowledge of statistics and machine learning (ML)*; *good writing skills*; (ideally) *completed coursework* at DIMA (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] NebulaStream: Complex Analytics Beyond the Cloud, https://www.nebula.stream/paper/zeuch_vliot.pdf
[2] Monitoring of Stream Processing Engines Beyond the Cloud: An Overview, https://www.nebula.stream/paper/chatziliadis_vliot2021.pdf
[3] Cheng, Ruosi & Wang, Yu. (2018). A Survey on Network Coordinate Systems. MATEC Web of Conferences. 232. 01037. 10.1051/matecconf/201823201037.
[4] Dabek, F., Cox, R., Kaashoek, F., & Morris, R. (2004, August). Vivaldi: A decentralized network coordinate system. In ACM SIGCOMM Computer Communication Review (Vol. 34, No. 4, pp. 15-26). ACM.
[5] Costa, M., Castro, M., Rowstron, R., & Key, P. (2004). PIC: Practical Internet coordinates for distance estimation. In Distributed Computing Systems, 2004. Proceedings. 24th International Conference on (pp. 178-187). IEEE.
[6] Ng, T. E., & Zhang, H. (2004, June). A Network Positioning System for the Internet. In USENIX Annual Technical Conference, General Track (pp. 141-154).
[7] Liu, Y., Du, H., & Ye, Q. (2015). WDCS: A Weight-Based Distributed Coordinate System. In Combinatorial Optimization and Applications.

20. Query Optimization in Distributed Stream Processing Systems

Advisor: Ankit Chaudhary

Appropriate Target Level: B.Sc., M.Sc.

Keywords: query optimization and operator placement in distributed stream processing systems

Description: The processing of geo-distributed data streams is a key challenge for many applications, such as the Internet of Things (IoT). Cloud-based stream processing engines (SPE) process data centrally and thus require all data to be present in the cloud in the cloud prior to processing them. However, this centralized approach becomes a bottleneck, when processing data from millions of geo-distributed sensors in a large-scale IoT infrastructure. Currently, a next generation data management system for the IoT is under development called *NebulaStream* (www.nebula.stream), which aims to mitigate the bottleneck using a decentralized approach involving fog devices. A major challenge for a SPE in this unified fog-cloud environment is the need to execute millions of queries concurrently. *NebulaStream* should be able to optimize and deploy incoming queries at high speed. Currently, we are investigating how to efficiently place operators for a large number of queries in a geo-distributed unified fog-cloud environment. Furthermore, we are conducting research to mitigate the effect of transient failures as well as monitor the effect that running queries have on system performance.

Prerequisites: *strong programming skills* preferably in C++; *good writing skills*; *a good understanding of query optimizers in database management systems*; (ideally) completed coursework in DIMA (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA) or possess the equivalent knowledge in distributed systems, stream processing engines, and databases.

21. Algorithmic Enhancements for the End-to-End Management of Large State in Distributed Stream Processing Engines

Advisor: Bonaventura Del Monte

Appropriate Target Level: M.Sc.

Keywords: state management for distributed data stream processing

Description: We seek to develop novel algorithms to enhance the end-to-end management of large state in distributed stream processing engines. Among the system characteristics (treated as first-class citizens) of interest are *resource elasticity*, *fault tolerance*, *load balancing*, *robust execution of stateful query*, and the *optimization of query plans*.

Concrete Thesis Topics:

1. **Exploring Lightweight Fault Tolerance for IoT Data Stream Processing**
   IoT infrastructures consist of thousands of devices connected over unreliable networks. To support stream processing workloads on an IoT infrastructure, systems require *fault-tolerance* to ensure consistency in the results. In this thesis, students will need to develop novel protocols for lightweight fault-tolerance and thereby enable reliable distributed stream processing. As a starting point, the Chandy-Lamport algorithm [1] for snapshotting a distributed system, and *mergeable replicated data types* [2], in the context of stateful stream processing will be examined.
   Expected Outcome: The design, implementation, and benchmarking of distributed protocols to enable lightweight fault-tolerance and support stateful stream processing

2. **Efficient Storage Backends for Stream Processing**
   Stream processing pipelines usually store state in a key/value data structure (hashmap). In this thesis, students will evaluate current hashmap implementations for purely *in-memory* as well as *disk-based* implementations.
   Expected Outcome: The *design of a benchmark for* as well as *an implementation of* multiple storage backends in *NebulaStream*, and the evaluation of the system across varying workloads. This thesis will be co-advised with Philipp Grulich.

Prerequisites: strong programming skills in C++; good writing skills; a good understanding of distributed database systems as well as stream processing systems; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] *Distributed Snapshots: Determining Global States of Distributed Systems*, http://lamport.azurewebsites.net/pubs/chandy.pdf.

[2] *Mergeable Replicated Data Types*, http://shorturl.at/cvHJL.

22. Enhancing Interoperability in Polystore Systems

Advisor: Haralampos Gavriilidis

Appropriate Target Level: B.Sc., M.Sc.

Keywords: data federation, interoperability, query languages, query optimization, polystores.

Description: Data stacks today are comprised of many data *storage* and *processing systems* [1]. To gain valuable insights, data scientists need to integrate and analyze all of the available data. To that end, data scientists either *implement custom data pipelines* [2] or *utilize existing hybrid analytic solutions* (i.e., *polystores*) [3-7]. Some of these approaches mirror concepts seen in federated databases [8,9], however, they go beyond the relational model. When composing custom data pipelines, users are forced to implement multiple queries and target them to each data management system (DMS) individually, which they subsequently need to assemble, and manually schedule. Although this approach enables each system to be exploited and optimized individually, to achieve optimal performance, it has some drawbacks. In particular, users need to be experts in all of the query languages and possess knowledge about the internals of each DMS. Additionally, users need to efficiently assemble the intermediate query results using hand-written ETL processes, which decreases the productivity and overall pipeline performance. To overcome these limitations, polystore systems offer middleware that mediates between users and multiple DMSs. Users would then implement declarative polystore queries and these would be transparently optimized and forwarded to each individual DMS. The unified interface frees users from having to query each source individually, and hence improves productivity. However, the performance of polystore queries largely depends on the *polystore optimization strategy* employed. Furthermore, since the expressivity of each polystore query language is limited, the set of supported polystore queries is equally limited.

Prerequisites: *good programming skills* in Java, Scala, and SQL; *good writing skills*; *experience using big data frameworks*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA); *interest in compilers*.

References

[1] Stonebraker, M., & Çetintemel, U. (2018). *"One size fits all" an idea whose time has come and gone*, Making Databases Work: The Pragmatic Wisdom of Michael Stonebraker (pp. 441-462).

[2] https://airflow.apache.org/

[3] Simitsis, A., Wilkinson, K., Castellanos, M., & Dayal, U. (2009, June). *QoX-driven ETL design: reducing the cost of ETL consulting engagements*, Proceedings of the 2009 ACM SIGMOD International Conference on the Management of Data (pp. 953-960).

[4] Xu, L., Cole, R. L., & Ting, D. (2019, July). *Learning to optimize federated queries*, Proceedings of the 2nd International Workshop on Exploiting Artificial Intelligence Techniques for Data Management @SIGMOD (pp. 1-7).

[5] Agrawal, D., Chawla, S., Contreras-Rojas, B., Elmagarmid, A., Idris, Y., Kaoudi, Z., ... & Papotti, P. (2018). *RHEEM: enabling cross-platform data processing: may the big data be with you!*, Proceedings of the VLDB Endowment, 11(11), 1414-1427.

[6] Duggan, J., Elmore, A. J., Stonebraker, M., Balazinska, M., Howe, B., Kepner, J., ... & Zdonik, S. (2015). *The bigdawg polystore system*, ACM SIGMOD Record, 44(2), 11-16.

[7] Sethi, R., Traverso, M., Sundstrom, D., Phillips, D., Xie, W., Sun, Y., ... & Berner, C. (2019, April). *Presto: SQL on everything*, 2019 IEEE 35th International Conference on Data Engineering (ICDE) (pp. 1802-1813).

[8] Hsiao, D. K. (1992). Fe*derated databases and systems: part i—a tutorial on their data sharing*, The VLDB Journal, 1(1), 127-179.

[9] Chawathe, S., Garcia-Molina, H., Hammer, J., Ireland, K., Papakonstantinou, Y., Ullman, J., & Widom, J. (1994). *The TSIMMIS project: Integration of heterogeneous information sources*.

23. Efficiently Embedding High-level User-defined Functions in the NebulaStream IoT Data Management System

Advisor: Philipp Grulich

Appropriate Target Level: M.Sc.

Keywords: stream processing, query compilation, modern hardware, code generation, augmenting data processing, job efficiency using query compilation techniques

Description: Since modern data processing pipelines often employ many programming languages (e.g., Java, Scala, Python), it is paramount that a data processing engine efficiently execute high-level, user-defined functions (UDFs). Unfortunately, it is difficult to optimize UDFs given that they contain arbitrary code. To address this problem, GraalVM, a just-in-time compiler that generates both *machine code* as well as an *LLVM* [2] *intermediate representation* is proposed. In this thesis, you will evaluate *GraalVM* [1] as a runtime for data processing workloads and investigate alternative embedding techniques for the efficient combination of a C++ based query engine (e.g., in NebulaStream) and Graal.

Expected Outcomes:

- investigate novel ways to embed high-level UDFs into NebulaStream,
- prototype a connector to the GraalVM LLVM backend,
- evaluate the implementation and conduct a performance analysis.

Prerequisites: *excellent programming skills in Java and C++; good writing skills*; *basic knowledge in compiler theory and LLVM*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] GraalVM: A High-performance Multilingual Runtime, https://www.graalvm.org/.
[2] LLVM Compiler Infrastructure, http://llvm.org/.

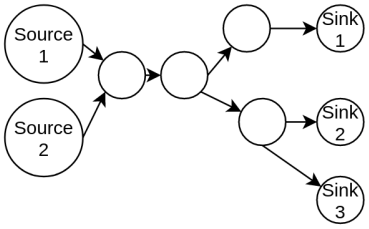
24. Code Generation for Complex Query Plans

Advisor: Philipp Grulich

Appropriate Target Level: M.Sc.

Keywords: stream processing, query compilation, modern hardware, code generation, augmenting data processing, job efficiency using query compilation techniques

Description: Code generation/query compilation is a well known technique that is used to improve the efficiency of modern data processing systems. In this thesis, this technique will be applied to *complex query graphs* (comprised of numerous source and sink operators), which arise in the optimization of merge query plans (e.g., in NebulaStream).



Expected Outcome:

- develop novel approaches to translate merge queries into code,
- devise optimization strategies based on the costs of queries,
- evaluate the implementation and conduct a performance analysis.

Prerequisites: *programming skills in C++*; *good writing skills*; *basic knowledge in data processing on modern hardware;* (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Grulich, Philipp M., et al. "Grizzly: Efficient Stream Processing Through Adaptive Query Compilation." Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. 2020.

[2] Neumann, Thomas. "Efficiently compiling efficient query plans for modern hardware." Proceedings of the VLDB Endowment 4.9 (2011): 539-550.

25. Adaptive Query Execution in NebulaStream

Advisor: Philipp Grulich

Appropriate Target Level: M.Sc.

Keywords: stream processing, query compilation, modern hardware, code generation, augmenting data processing, job efficiency using query compilation techniques

Description: In the Grizzly paper [1], we proposed an adaptive query execution technique that enables a stream processing system to react to changes in data characteristics at runtime. In this thesis, we want to implement this technique in the NebulaStream system, and explore the design space of potential optimizations.

Expected Outcome:

- implement adaptive query compilation for the NebulaStream system,
- devise optimization strategies based on the runtime characteristics,
- evaluate the implementation and conduct a performance analysis.

Prerequisites: *programming skills in C++*; *good writing skills*; *basic knowledge in data processing on modern hardware;* (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Grulich, Philipp M., et al. "Grizzly: Efficient Stream Processing Through Adaptive Query Compilation." Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. 2020.

26. Data Stream Summarization Using Custom Hardware (FPGAs)

Advisor: Martin Kiefer

Appropriate Target Level: B.Sc., M.Sc.

Keywords: approximate data analysis using modern hardware, energy efficient computing

Description: My research is centered on reducing the time it takes to conduct data analysis. To achieve this, we seek to leverage modern hardware architectures and employ approximate computing techniques (e.g., data sketching [1]), which is faster and more efficient.

Power-efficient data analysis is an increasingly important problem in the era of big data: The amount of available data continues to increase exponentially and for economic and environmental reasons, we need to ensure that the energy demands required to analyze the data do not grow exponentially as well. In this thesis, we seek to analyze data streams using stream summarization techniques and custom hardware on FPGAs, while taking energy efficiency into consideration.

Prerequisites: *programming skills* in C/C++, OpenCL, VHDL, or Python; *good writing skills*; (preferably) *interest in modern hardware*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DMH / formerly MHD, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Graham Cormode. 2017. Data Sketching: The approximate approach is often faster and more efficient. In *ACM Queue*, Vol. 15, Issue 2. https://queue.acm.org/detail.cfm?id=3104030

27. Improving Query Optimization Using Modern Hardware

Advisor: Martin Kiefer

Appropriate Target Level: B.Sc., M.Sc.

Keywords: approximate data analysis using modern hardware

Description: My research is centered on *reducing the time it takes to conduct data analysis*. To achieve this, we seek to leverage modern hardware architectures and employ approximate computing techniques (e.g., data sketching [1]), which is faster and more efficient.

The query optimizer is at the heart of state-of-the-art relational database systems. It derives different execution plans for a given query and selects the cheapest one based on statistics and a cost model. However, this has to be done in a very tight time budget, since query optimization delays query execution. In this thesis, we seek to leverage modern hardware and algorithmics to optimize queries more efficiently. In particular, using bandwidth-optimized kernel density models as learning selectivity estimators on GPUs, to improve the statistics available to the query optimizer.

Prerequisites: *programming skills* in C/C++, OpenCL, VHDL, or Python; *good writing skills*; (preferably) *interest in modern hardware*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Graham Cormode. 2017. Data Sketching: The approximate approach is often faster and more efficient. In *ACM Queue*, Vol. 15, Issue 2. https://queue.acm.org/detail.cfm?id=3104030

28. Dynamic Checkpointing for the NebulaStream IoT Platform

Advisor: Anastasiia Kozar

Appropriate Target Level: M.Sc.

Keywords: data stream processing, fault-tolerance, sensors, IoT

Description: The rise of sensor-produced data introduces a new class of applications for low-latency and high-throughput analytics. The data is produced in a distributed manner and streamed through multiple IoT devices. Due to the high network connectivity cost, the data processing is offloaded to the intermediate nodes instead of getting sent to remote destinations. These nodes are highly-distributed and communicate using a wireless connection organized across three computing types: the edge, fog, and cloud [2]. To enable reliable processing within these three computing types is a daunting task, one that requires devising a conceptually unified environment that can handle the runtime dynamics. One of the solutions will be to periodically checkpoint [1] the state of every node in the system and save it to remote persistent memory. The question arises, "How often do we want to send the state of a node? *every time it changes*? *in a given time period*? *after every x tuples*? This thesis aims to define the connection between the load of a node and the state submission rate, so as to enable the fault-tolerance technique (running in the background) to be more agile. The goal is to devise an algorithm that will dynamically adjust checkpointing frequency to the node load.

Prerequisites: *strong programming skills* (preferably in C++); *good writing skills*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DMH / formerly MHD, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] *High-Availability Algorithms for Distributed Stream Processing*, Jeong-Hyon Hwang, Magdalena Balazinska, Alexander Rasin, Ugur Cetintemel, Michael Stonebraker, and Stan Zdonik (2005)

[2] Zeuch, S., Zacharatou, E.T., Zhang, S., Chatziliadis, X., Chaudhary, A., Monte, B.D., Giouroukis, D., Grulich, P.M., Ziehn, A., & Mark, V. (2020). *NebulaStream: Complex Analytics Beyond the Cloud*. Open J. Internet Things, 6, 66-81.

29. An Exploration of the Compatibility of Fault Tolerance Techniques During Query Merging

Advisor: Anastasiia Kozar

Appropriate Target Level: B.Sc.

Keywords: data stream processing, fault-tolerance, IoT, query processing, query merging

Description: Current Internet of Things (IoT) systems consist of billions of sensors that generate voluminous amounts of data in the form of data streams. If we consider the current IoT hardware capabilities and the sheer amount of data, data processing is done locally on the devices without sending data to remote servers. Data stream engines based on the IoT unify computing infrastructures, such as the fog, cloud computing, and sensor networks [2], thereby enabling a comprehensive set of possibilities for decentralized data processing. In such systems, operators are deployed to *worker* nodes, and data that belongs to a given user query runs along predefined paths. Whereas some of the user queries are divergent, some may be partially or fully identical. In the second case, these queries are merged, in order to optimize the system throughput. Yet, for every query, a user can specify the fault-tolerance guarantee [1] that she wants to achieve. Such processing guarantees can be strong, for example, *exactly-once* or relaxed, like *at-least-once* or *at-most-once*. The goal of this thesis is to investigate how different fault tolerance guarantees can be matched when merging queries.

Prerequisites: *good writing skills*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DMH / formerly MHD, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] *High-Availability Algorithms for Distributed Stream Processing*, Jeong-Hyon Hwang, Magdalena Balazinska, Alexander Rasin, Ugur Cetintemel, Michael Stonebraker, and Stan Zdonik (2005)

[2] Zeuch, S., Zacharatou, E.T., Zhang, S., Chatziliadis, X., Chaudhary, A., Monte, B.D., Giouroukis, D., Grulich, P.M., Ziehn, A., & Mark, V. (2020). *NebulaStream: Complex Analytics Beyond the Cloud*. Open J. Internet Things, 6, 66-81.

30. Scalable GPU Co-processing with Fast, Cache-coherent Interconnects

Advisor: Clemens Lutz

Appropriate Target Level: B.Sc., M.Sc.

Keywords: scalable GPU co-processing, cache-coherent interconnects

Description: GPUs and other modern processors are capable of very fast data processing. For example, a high-end Nvidia GPU is capable of reading from its built-in memory at a rate of 900 GB/s, and compute 14 trillion floating-point operations per second (i.e., 14 TFLOPS). This is more than 20 times faster than regular CPUs. Our aim is to leverage this increased processing power, in order to analyze data more efficiently. In particular, devising novel ways to *access data from a GPU more efficiently* or *execute a SQL JOIN operator faster*.

Prerequisites: *programming skills in* C/C++, CUDA, or OpenCL; *good writing skills*; *performance-oriented programming of CPUs and GPUs*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DMH / formerly MHD, DBPRA, DBPRO, DBSEM, ISDA).

31. Data Compression in Secure Join Processing

Advisor: Kajetan Maliszewski

Appropriate Target Level: M.Sc.

Keywords: confidential computing, join processing, compression, Intel SGX

Description: More and more companies are shifting towards the public cloud. It has been shown that the maintenance of a private infrastructure is costly and ill-suited to many. However, many organizations that work with sensitive data (e.g., hospitals, financial institutions) are unable to use a public cloud. These entities need absolute data confidentiality, which the public cloud providers do not grant. In recent years, hardware manufacturers proposed *Trusted Execution Environments* (TEEs) as a solution to this problem. This novel secure hardware promises to work on encrypted code and data at all times. This allows users to securely execute their code on remote machines, (e.g., in the public cloud). However, TEEs introduce some challenges. One of these challenges is the availability of very small memory that can be securely used by algorithms. In general, we have observed that TEEs do not handle data-heavy applications well, such as DBMS. Working with various join algorithms, we have identified the need for a complete redesign of the DBMS stack. A possible solution would be to store and process compressed data [3]. In this thesis, you will seek to improve the performance of join operators using various compression techniques. You will get to know the common techniques for data compression in databases and use them in join processing. You will further propose your own compression technique optimized for the Intel SGX [1, 2], a TEE implementation.

Prerequisites: *programming skills in C/C++*; *good writing skills*; *basic understanding about how CPUs and main memory work (*e.g., cache, transaction lookaside buffer, memory control); *knowledge about SQL joins*; (ideally) *completed several DIMA* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Victor Costan and Srinivas Devadas. 2016. Intel SGX Explained. International Association for Cryptologic Research. https://eprint.iacr.org/2016/086.pdf.

[2] Intel® Software Guard Extensions. https://intel.ly/3bPhxaT.

[3] Lee, Jae-Gil, Gopi Attaluri, Ronald Barber, Naresh Chainani, Oliver Draese, Frederick Ho, Stratos Idreos, et al. “Joins on Encoded and Partitioned Data.” *Proceedings of the VLDB Endowment* 7, no. 13 (August 1, 2014): 1355–66.

32. Secure Relational Operators with Encryption and TEEs

Advisor: Kajetan Maliszewski

Appropriate Target Level: B.Sc., M.Sc.

Keywords: confidential computing, data encryption, Intel SGX

Description: Trusted Execution Environment (TEEs) [1, 2] are a popular tool for protecting data in-use. However, TEEs were not designed for data-intensive applications. They operate on small memory (i.e., around 100 MB) and impose high processing costs for various operations (e.g., interaction with the OS). An alternative to TEEs is to process encrypted data. For example, CryptDB [4] mixes several encryption schemes to protects users data – deterministic encryption (DET), order-preserving encryption (OPE), random encryption (RND), and homomorphic encrytpion (HOM). These techniques do not require any specialized hardware (i.e., they are purely-software techniques), but they might introduce extremely high processing costs [3]. In this thesis, you will combine these two approaches – TEEs and data encryption. You will devise new algorithms for relational operators that do part of the processing on encrypted data in the CPU and the other part using secure hardware (i.e., TEEs). First, you will need to familiarize yourself with encryption schemes used in DBMS (e.g., DET, OPE, HOM) and TEEs. Second, you will devise an algorithm for a relational operator of your choice (e.g., join, aggregate, filter) that uses both techniques. Third, you will implement and benchmark your solution.

Prerequisites: *programming skills in C/C++*; *good writing skills*; *basic understanding about how CPUs and main memory work (*e.g., cache, transaction lookaside buffer, memory control); *knowledge about SQL joins*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Victor Costan and Srinivas Devadas. 2016. Intel SGX Explained. International Association for Cryptologic Research. https://eprint.iacr.org/2016/086.pdf.

[2] Intel® Software Guard Extensions. https://intel.ly/3bPhxaT.

[3] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. 2011. Can homomorphic encryption be practical? In Proceedings of the 3rd ACM workshop on Cloud computing security workshop. 113–124.

[4] Raluca Ada Popa, Catherine MS Redfield, Nickolai Zeldovich, and Hari Balakrishnan. 2011. CryptDB: Protecting confidentiality with encrypted query processing. In Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. 85–100.

33. High Level Abstraction Layers for Data Processing on Heterogeneous CPU-GPU Architectures

Advisor: Dwi Prasetyo Adi Nugroho

Appropriate Target Level: M.Sc.

Keywords: GPU, query execution, programming interface

Description: To accelerate data processing, it is commonplace today to exploit both CPUs and GPUs. However, there is a drawback: CPUs and GPUs each employ their own programming framework. Unfortunately, this increases the complexity in developing a data processing platform. To mitigate this problem, a natural solution would be to reuse the same code base, and compile the code for both CPUs and GPUs. Recent studies in the HPC domain [1,2] have shown how existing high-level abstraction frameworks (e.g., SYCL[2], OpenCL[3], Kokkos[4]) can help. In this thesis, we will investigate the challenges and opportunities that high-level abstraction layers offer to accelerate data processing.

Prerequisites: *programming skills in C/C++*; *good writing skills*; *basic understanding of GPU architectures and parallel execution*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DMH / formerly MHD, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Homerding, Brian, and John Tramm. “Evaluating the Performance of the hipSYCL Toolchain for HPC Kernels on NVIDIA V100 GPUs.” Proceedings of the International Workshop on OpenCL. 2020.

[2] Deakin, Tom, and Simon McIntosh-Smith. “Evaluating the performance of HPC-style SYCL applications.” Proceedings of the International Workshop on OpenCL. 2020.

[2] SYCL is a cross-platform abstraction layer that enables code for heterogeneous processors to be written using standard ISO C++ with the host and kernel code for an application contained in the same source file.
[3] OpenCL (Open Computing Language) is a framework for writing programs that execute across heterogeneous platforms consisting of CPUs and GPUs, among other processors or hardware accelerators.
[4] Kokkos is a templated C++ library that provides abstractions to allow a single implementation of an application kernel to run efficiently on different kinds of hardware, such as GPUs, Intel Xeon Phis, or many-core CPUs.

34. Scaling Streaming Graph Neural Networks

Advisor: Serafeim “Makis” Papadias

Appropriate Target Level: M.Sc.

Keywords: machine learning, representation learning, streaming graphs

Description: Various real-world applications are modeled as *graphs*. For example, social networks are large graphs, where the nodes and edges represent the users and their friendships, respectively. Today, there is great interest in performing machine learning (ML) on graphs. A typical ML application is *link prediction*, which seeks to predict new friendships in social media. Many real-world graphs are inherently dynamic, i.e., new nodes and edges are constantly being added or existing ones deleted. Unfortunately, most *graph neural network* models are unable to harness dynamic data, which can boost the performance of many ML tasks. Recently, a new Dynamic Graph Neural Network (DGNN) was proposed by Ma et al. [1], which is capable of incorporating dynamic data. Although the proposed approach is effective, it is only suitable for small graphs. In this thesis, the goal is to develop a scalable, dynamic graph neural network model for data streams.

Prerequisites: strong *programming skills (e.g., Java)*; *good writing skills*; *knowledge in database systems and machine learning;* (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Yao Ma et al. Streaming Graph Neural Networks. In *SIGIR 2020.*

35. The Management of Data Science Processes

Advisor: Sergey Redyuk

Appropriate Target Level: B.Sc., M.Sc.

Keywords: machine learning pipelines, experiment databases, data science processes

Description: The development of modern data science pipelines is an iterative process that captures the specification of the data analytics to be executed. Among the components in these pipelines are the ability to *profile data*, support for *exploratory data analysis*, *data preprocessing* and *feature engineering* capabilities, as well as solutions for *model selection* and *hyperparameter tuning*, *performance evaluation*, *visualization*, and *reporting*. Many of the algorithms employed in these components operate under implicit assumptions that neither novice users nor domain experts are aware of. Moreover, frameworks that support pipeline abstractions (e.g., Scikit-learn, SparkML) typically do not provide optimization strategies for efficient pipeline execution. As a result, creating and executing data science pipelines (represented as complex graphs, comprised of *data manipulation* and *machine learning* operations) efficiently is challenging. Among the challenges to be overcome in the end-to-end management of data science processes, there are several thesis topics: (i) the *declarative specification of data science processes* [1] (e.g., to express complex data science processes in a unified way and enable the systematic comparison of processes and factor analysis), (ii) *experiment databases* [2] (e.g., that store previously executed data science pipelines and enable new ones to be improved), and (iii) the development of *optimization rules* [3], to enable the efficient execution of end-to-end data science pipelines.

Prerequisites: *programming skills in Python*; *good writing skills*; *experience with the Python data science toolkit* (e.g., pandas, numpy, scikit-learn, keras)*;* (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA); (ideally) *prior experience or completed coursework in machine learning*.

References

[1] S. Redyuk. Automated Documentation of End-to-End Experiments in Data Science. In Ph.D. Symposium track, IEEE 35th International Conference on Data Engineering (ICDE'19)

[2] Van Rijn et al. OpenML: A collaborative science platform. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases 2013 (pp. 645-649). Springer, Berlin, Heidelberg.

[3] Derakhshan et al. Optimizing Machine Learning Workloads in Collaborative Environments. In: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data.

36. Large-Scale Machine Learning

Advisor: Alexander Renz-Wieland

Appropriate Target Level: B.Sc., M.Sc.

Keywords: large-scale machine learning, parameter servers

Description: It is advantageous to train large-scale machine learning (ML) models on a cluster (instead of a single machine), since it increases the amount of available compute power and memory. However, the drawback is that this approach requires communication among the cluster nodes, in order to synchronize model parameters. For some ML models, this synchronization step is the dominating part of the training process. In this case, it would not be prudent to increase the number of nodes, since this would result in a drop in the performance. To reduce the communication overhead, researchers have developed algorithms to create and exploit parameter locality (e.g., stemming from the training algorithm, ML model, or training data), where each worker updates a subset of the model parameters at a given point in time. These subsets typically change (i.e., workers update different subsets) throughout training process. ML developers typically need to implement locality-exploiting algorithms (LEAs) from scratch, which requires them to possess in-depth knowledge (i.e., low level details) about distributed computing environments. We are developing a system that enables researchers and practitioners to implement LEAs, without the need for detailed distributed computing knowledge. We seek to make parameter servers (the state-of-the-art architecture for distributed ML) usable and more efficient for LEAs. Several thesis topics in this subfield arise, including addressing system-related challenges as well as experimenting with the system along specific ML models.

Prerequisites: *programming skills in C++*; *good writing skills*; (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA); (ideally) *prior experience or completed coursework in machine learning and distributed systems*.

37. Data Processing on Heterogeneous CPU/GPU Systems

Advisor: Viktor Rosenfeld

Appropriate Target Level: B.Sc., M.Sc.

Keywords: exploiting GPU hardware for query processing, scheduling query processing tasks on heterogeneous processors

Description: GPUs are powerful co-processors that can be used to speed up query processing. In this context, we offer two thesis topics.

Topic 1: Recent dedicated GPUs contain specialized hardware cores to accelerate matrix computations. This new hardware makes GPUs interesting processors to exploit machine learning for query optimization. In this thesis, you would investigate *which query optimization tasks can exploit this hardware* effectively and *what benefit it provides on query performance*.

Topic 2: On CPUs, the state-of-the-art query processing models are *query compilation* and *vectorization*. They have comparable performance, however, which one is faster depends on the specific query. These processing models have been implemented on GPUs, but it is not yet clear, which one is faster. Due to the cache architecture and high launch overheads, vectorization is more difficult to efficiently implement on GPUs than on CPUs. In this thesis, you would investigate *how to implement vectorization efficiently* and *conduct a detailed comparison with query compilation*.

Prerequisites: *strong programming skills in C/C++, OpenCL, CUDA*; *good writing skills*; *interest in low-level programming*, *processor architectures;* (ideally) *completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DMH / formerly MHD, DBPRA, DBPRO, DBSEM, ISDA).

38. Large-Scale Complex Event Processing for the Internet of Things

Advisor: Ariane Ziehn

Appropriate Target Level: B.Sc., M.Sc.

Keywords: large-scale complex event processing, stream processing, distributed systems

Description: Complex event processing (CEP) is a common method used to detect events in data streams and trigger actions correspondingly whenever a pattern of interest is observed. Ideally, such solutions perform these actions in real-time using a scalable stream processing system. User-defined rules specify both, the events and their composition, as well as the actions that enable autonomous real-time decision making in a wide range of existing and emerging applications (e.g., live maps, smart street lamps, vehicle pollution control). However, the state-of-the-art CEP engines are still mainly based on central architectures and serial processing models, which prevents low-latency and the real-time processing of Big Data streams. Due to an ever-increasing number of IoT devices, their geographical distribution, and their environmental heterogeneity, the limitations of CEP engines become even more critical. Our objective is to explore how to overcome current bottlenecks of CEP to enable large-scale CEP in distributed stream processing systems.

General Topics: This topic is related to my current research involving a next-generation data management platform for the IoT called NebulaStream [3], which is currently under development in the DIMA Group. (Students are also at liberty to propose their own topic related to my work. For details, ideas, and recent research questions, read my Ph.D. workshop paper [1] and the latest CEP survey [2]). The scope of the planned thesis will be based on your skills and interests.

Prerequisites: *programming skills in C++*; *good writing skills*; *knowledge of distributed systems* (particularly, stream processing engines); (ideally) *prior experience or completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Ziehn, Ariane. "Complex Event Processing for the Internet of Things." *fog* 1.3: 4. http://ceur-ws.org/Vol-2652/paper01.pdf.

[2] Giatrakos, Nikos, et al. "Complex event recognition in the big data era: a survey." *The VLDB Journal* 29.1 (2020): 313-352.

[3] The NebulaStream System for the Management of IoT Data. https://www.nebula.stream/.

39. A Standardized Complex Event Processing Benchmark for Big Data Platforms

Advisor: Ariane Ziehn

Appropriate Target Level: B.Sc., M.Sc.

Keywords: large-scale complex event processing, stream processing, distributed systems

Description: Complex event processing (CEP) is a common method used to detect events in data streams and trigger actions correspondingly whenever a pattern of interest is observed. Ideally, such solutions perform these actions in real-time using a scalable stream processing system. Via user-defined rules one can specify the *events* (and their composition) as well as the *actions*, to enable autonomous real-time decision making in a wide range of existing and emerging applications (e.g., live maps, smart street lamps, vehicle pollution control). Standardized benchmarks for CEP do not yet exist due to the absence of a widely accepted formal framework and terminologies [1]. State-of-the-art (SOTA) CEP engines provide neither *a single common pattern specification language,* nor *a uniform evaluation mechanism*, which makes benchmarking non-trivial. As a consequence, there is a need to *define the key criteria* for Big Data [1]. Furthermore, SOTA CEP engines are still mainly based on central architectures and serial processing models. Unfortunately, commonly used metrics are ill-suited for CEP in Big Data settings. As a result, suitable metrics must be carefully chosen. In this thesis, the goals are to: (1) *create a CEP benchmarking platform prototype*, to facilitate evaluating CEP engines and determining their strengths and weaknesses (using alternative approaches, like TPC-H [6], Yahoo! Streaming Benchmark [4], Karimov et al. [3], and those discussed in Section 3 of the CEP survey [1]), and (2) *conduct an in-depth analysis* of select CEP engines.

Prerequisites: *programming skills; good writing skills*; *experience with distributed systems* (particularly, stream processing engines); *knowledge of fair* benchmarking; (ideally) *prior experience or completed several DIMA courses* (e.g., SDS / formerly AIM-3, BDSPRO / formerly BDAPRO, BDASEM, DBT, DBTLAB / formerly DBT-PRA, DBPRA, DBPRO, DBSEM, ISDA).

References

[1] Giatrakos, Nikos, et al. “Complex event recognition in the big data era: a survey.“ *The VLDB Journal* 29.1 (2020): 313-352.

[2] Ziehn, Ariane. ”Complex Event Processing for the Internet of Things.“ *fog* 1.3: 4. (http://ceur-ws.org/Vol-2652/paper01.pdf)

[3] Karimov, Jeyhun, et al. „Benchmarking Distributed Stream Data Processing Systems.“ *arXiv preprint arXiv:1802.08496* (2018).

[4] Raasveldt, Mark, et al. “Fair benchmarking considered difficult: Common pitfalls in database performance testing.“ *Proceedings of the Workshop on Testing Database Systems*. 2018.

[5] S. Chintapalli, D. Dagit, B, et al., Benchmarking streaming computation engines: Storm, Flink and Spark Streaming. In IPDPSW, pages 1789–1792, 2016.
https://github.com/yahoo/streaming-benchmarks

[6] TPC-H is a Decision Support Benchmark. http://www.tpc.org/tpch/