Report from Dagstuhl Seminar 17101

Databases on Future Hardware

Edited by

Gustavo Alonso[1], Michaela Blott[2], and Jens Teubner[3]

1 **ETH Zürich, CH,** `alonso@inf.ethz.ch`
2 **Xilinx – Dublin, IE,** `michaela.blott@xilinx.com`
3 **TU Dortmund, DE,** `jens.teubner@cs.tu-dortmund.de`

Abstract

A number of physical limitations mandate radical changes in the way how we build computing hard- and software, and there is broad consensus that a stronger interaction between hard- and software communities is needed to meet the ever-growing demand for application performance.

Under this motivation, representatives from various hard- and software communities have met at the Dagstuhl seminar "Databases on Future Hardware" to discuss the implications in the context of database systems. The outcome of the seminar was not only a much better understanding of each other's needs, constraints, and ways of thinking. Very importantly, the group identified topic areas that seem key for the ongoing shift, together with suggestions on how the field could move forward. During the seminar, it turned out that the future of databases is not only a question of technology. Rather, economic considerations have to be taken into account when building next-generation database engines.

Seminar March 5–10, 2017 – http://www.dagstuhl.de/17101
1998 ACM Subject Classification C.0 Computer Systems Organization, H.2.4 Database Management Systems
Keywords and phrases computer architecture, hardware support for databases, non-volatile
Digital Object Identifier 10.4230/DagRep.7.3.1

1 Executive Summary

Jens Teubner

Computing hardware is undergoing radical changes. Forced by physical limitations (mainly *heat dissipation* problems), systems trend toward *massively parallel* and *heterogeneous* designs. New technologies, *e.g.*, for *high-speed networking* or *persistent storage* emerge and open up new opportunities for the design of database systems. This push by technology was the main motivation to bring top researchers from different communities – particularly hard- *and* software – together to a Dagstuhl seminar and have them discuss about "Databases on Future Hardware." This report briefly summarizes the discussions that took place during the seminar.

With regards to the mentioned *technology push*, during the seminar *bandwidth*; *memory and storage technologies*; and *accelerators* (or other forms of specialized computing functionality or instruction sets) were considered the most pressing topic areas in database design.

But it turned out that the field is influenced also by a strong push from *economy/market*. New types of *applications* – in particular Machine Learning – as well as the emergence of

Databases on Future Hardware, *Dagstuhl Reports*, Vol. 7, Issue 3, pp. 1–18
Editors: Gustavo Alonso, Michaela Blott, and Jens Teubner

Dagstuhl Reports
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

"compute" as an independent type of resources – *e.g.*, in the form of *cloud computing* or *appliances* – can have a strong impact on the viability of a given system design.

Bandwidth; Memory and Storage Technologies

During the seminar, probably the most often stated issue in the field was *bandwidth* – at various places in the overall system stack, such as CPU ↔ memory; machine ↔ machine (network); access to secondary storage (*e.g.*, disk, SSD, NVM). But very interestingly, the issue was not only brought up as a key limitation to database performance by the seminar attendees with a software background. Rather, it also became clear that the hardware side, too, is very actively looking at bandwidth. The networking community is working at ways to provide more bandwidth, but also to provide hooks that allow the software side to make better use of the available bandwidth. On the system architecture side, new interface technologies (*e.g.*, NVlink, available in IBM's POWER8) aim to ease the bandwidth bottleneck.

Bandwidth usually is a problem only *between* system components. To illustrate, HMC memories ("hybrid memory cube") provide only 320 GB/s of external bandwidth, but internally run at 512 GB/s per cube ("vault"); in a 16-vault configuration, this corresponds to 8 TB/s of internal bandwidth. This may open up opportunities to build heterogeneous system designs with *near-data processing* capabilities. HMC memory units could, for instance, contain (limited) processing functionality associated with every storage vault. This way, simple tasks, such as data movement, re-organization, or scanning could be off-loaded and performed right where the data resides. Similar concepts have been used, *e.g.*, to filter data in the network, pre-process data near secondary storage, etc.

In breakout sessions during the seminar, attendees discussed the implications that such system designs may have. Most importantly, the designs will require to re-think the existing *(programming) interfaces*. How does the programmer express the off-loaded task? Which types of tasks can be off-loaded? What are the limitations of the near-data processing unit (*e.g.*, which memory areas can it access)? How do host processor and processing unit exchange tasks, data, and results? Clearly, a much closer collaboration will be needed between the hard- and software sides to make this route viable.

But new designs may also shake up the commercial market. The traditional hardware market is strongly separated between the memory and logic worlds, with different manufacturers and processes. Breaking up the separation may be a challenge both from a technological and from a business/market point of view.

The group found only little time during the seminar to discuss another potential game-changer in the memory/storage space. Companies are about to bring their first *non-volatile memory (NVM)* components to the market (and, in fact, Intel released its first round of "3D XPoint" products shortly after the seminar). The availability of cheap, high-capacity, byte-addressable, persistent storage technologies will have profound impact on database software. Discussions during the seminar revolved around the question whether classical persistent (disk-based) mechanisms or in-memory mechanisms are more appropriate to deal with the new technology.

Accelerators

A way of dealing with the technology trend toward heterogeneity is to enrich general-purpose systems with more specialized processing units, *accelerators*. Popular incarnations of this idea are *graphics processors (GPUs)* or *field-programmable gate arrays (FPGAs)*; but there

are also co-processing units for floating-point arithmetics, multimedia processing, or network acceleration.

Accelerators may fit well with what was said above. *E.g.*, they could be used as near-data processing units. But also the challenges mentioned above apply to many accelerator integration strategies. Specifically, the proper *programming interface*, but also the role of an accelerator in the software system stack – *e.g.*, sharing it between processes – seem to be yet-unsolved challenges.

During the seminar, also the role of accelerators specifically for database systems was discussed. It was mentioned, on the one hand, that accelerators should be used to accelerate functionality outside the database's core tasks, because existing hard- and software is actually quite good at handling typical database tasks. On the other hand, attendees reported that many of the non-core-database tasks, Machine Learning in particular, demand a very high flexibility that is very hard to provide with specialized hardware.

New Applications / Machine Learning

Databases are the classical device to deal with high volumes of data. With the success of Machine Learning in many fields of computing, the question arises how databases and Machine Learning applications should relate to one another, and to which extent the database community should embrace ML functionality in their system designs.

Some of the seminar attendees have, in fact, given examples of very impressive and successful systems that apply ideas from database co-processing to Machine Learning scenarios. In a breakout session on the topic, it was concluded that the two worlds should still be treated separately also in the future.

A key challenge around Machine Learning seems to be the very high expectations with regard to the flexibility of the system. ML tasks are often described in high-level languages (such as R or Python) and demand expressiveness that goes far beyond the capabilities of efficient database execution engines. Attempts to extend these engines with tailor-made ML operators were not very well received, because even the new operators were too restrictive for ML users.

Economic/Market Considerations

Somewhat unexpectedly, during the seminar it became clear that the interplay of databases and hardware is not only a question of technology. Rather, examples from the past and present demonstrate that even a technologically superior database solution cannot survive today without a clear business case.

The concept of *cloud computing* plays a particularly important role in these considerations. From a business perspective, compute resources – including database functionality – have become a commodity. Companies move their workloads increasingly toward cloud-based systems, raising the question whether the future of databases is also in the cloud.

A similar line of arguments leads to the concept of *database appliances*. Appliances package database functionality in a closed box, allowing *(a)* to treat the service as a commodity (business aspect) and *(b)* to tailor hard- and software of the appliance specifically to the task at hand, with the promise of maximum performance (technology aspect).

And, in fact, both concepts – cloud computing and appliances – may go well together. Cloud setups enable to control the entire hard- and software stack; large installations may provide the critical mass to include tailor-made (database) functionality also within the cloud.

2 Table of Contents

3 Overview of Talks

3.1 How to integrate FPGAs into data processing systems

Gustavo Alonso (ETH Zürich, CH)

The increasing amount of data and the growing complexity of workloads pose a significant challenge to existing data management systems. Data is growing not only in size but also in variety and heterogeneity. Workloads are more complex not only in regards to performance but also in terms of algorithmic heterogeneity and embedding into interactive systems. Hardware acceleration in general and FPGAs in particular are a credible first step towards addressing many of these challenges and doing so both form the performance perspective but also from an efficiency and functionality standpoint. In this talk I will cover our research journey over the last ten years in pursuit of FPGA based solutions for data processing and data management problems. The talk will follow the evolution of FPGAs as an architectural component in a computer system and illustrate the algorithmic, design, integration, and system issues that need to be solved before FPGAs become mainstream tools. For simplicity, and to keep the discussion focused, I will center the examples around DoppioDB, a relational database engine we have developed over HARP v1 combining a conventional, open source column store engine (MonetDB) with a cache coherent FPGA accelerator.

References

1 Zsolt István, David Sidler, Gustavo Alonso: Runtime Parameterizable Regular Expression Operators for Databases. FCCM 2016: 204–211
2 Kaan Kara, Gustavo Alonso: Fast and robust hashing for database operators. FPL 2016: 1–4
3 Louis Woods, Zsolt István, Gustavo Alonso: Ibex – An Intelligent Storage Engine with Support for Advanced SQL Off-loading. PVLDB 7(11): 963–974 (2014)

3.2 Scaling Neural Networks On Reconfiguable Logic

Michaela Blott (Xilinx – Dublin, IE)

The ongoing research on Neural Networks has started to focus on reducing the computation and storage requirements to make their deployment feasible in energy constraint compute environments. One of the promising opportunities is the reduction of the compute and storage down to a few bit precision whereby these networks achieve close to state of the art accuracy compared to their floating point counterparts. In this talk, we will show an automated framework for implementing these reduced precision (and in the extreme case fully binarized) neural networks on reconfigurable logic that can scale reduced precision neural networks onto an FPGA-based inference accelerator, given a set of fixed design constraints. We show, that the compute performance can scale well beyond typical floating point performance, currently demonstrating 10ks to millions of images per second for inference, 14 TOps compute performance with power consumption < 25W on today's devices. Results on the accuracy, architecture comparison to other approaches and detailed implementation of the latest large networks will also be presented.

3.3 Making the case for a closer DBMS and OS collaboration

Alexander Böhm (SAP SE – Walldorf, DE)

Performance optimization and the creation of highly efficient systems has always been a major focus of the database community. In academia, this is reflected by a vast number of publications that describe sophisticated techniques for all major components and aspects of a DBMS. Recently, the DBMS community broadened its scope towards hardware. This means no longer "just" looking on how to optimize the DBMS itself, but doing hardware/software co-design e.g. by exploiting vector instructions for efficient scans in in-memory databases such as HyPer or SAP HANA.

An important building block however is often ignored / heavily underestimated: The operating system. Modern, enterprise-class in-memory systems spend a lot of time re-implementing OS functionality in userspace, i.e. thread scheduling, memory management, and synchronization primitives, to only name a few.

We as a DBMS community need to pay more attention to the operating system and achieve a better integration instead of replicating and duplicating features and innovations the OS community creates.

3.4 Implications of Secure Computing Platforms for Databases

Ken Eguro (Microsoft Research – Redmond, US)

Main reference A. Arasu, K. Eguro, M. Joglekar, R. Kaushik, D. Kossmann, R. Ramamurthy, "Transaction processing on confidential data using cipherbase", in Proc. of the 31st Int'l Conf. on Data Engineering (ICDE), pp. 435–446, IEEE, 2015.
URL https://doi.org/10.1109/ICDE.2015.7113304
Main reference A. Arasu, S. Blanas, K. Eguro, R. Kaushik, D. Kossmann, R. Ramamurthy, R. Venkatesan, "Orthogonal Security with Cipherbase", in Proc. of the 6th Biennial Conf. on Innovative Data Systems Research (CIDR'13), Microsoft, 2013.
URL https://www.microsoft.com/en-us/research/publication/orthogonal-security-with-cipherbase/

Most DBMS are vulnerable to attack from administrators and hackers. This is because their security is largely limited to on-disk encryption and data is generally held in plaintext when loaded into RAM. While there have been attempts to help address this, by-in-large these solutions have had their own shortcomings. There are a number of up-and-coming secure computing platforms which can unlock new ways to address the security problem. In this talk we cover some of these hardware-based and software-based (but hardware assisted) platforms. We will also discuss the design space in which these platforms can be utilized by DBMS, how the SQL Server Always Encrypted feature will leverage these platforms, and open questions that remain, such as potential security/efficiency tradeoffs

3.5 Databases on Future Hardware

Peter Hofstee (IBM Research Lab. – Austin & TU Delft)

We discussed how scale-out systems are likely to develop in the near future. The most significant changes are likely to come from increases in network and SCM/NVM bandwidth that can rival that of memory in systems with reasonable cost. We discuss how this forces us to rethink the role of main memory and that we need a system concept that allows all components to interact without going through system DRAM. We explain the relatively new coherent attach interfaces on POWER8 and POWER9 and discuss where acceleration can make sense. We give a few database-related examples that can benefit from this new infrastructure: key-value stores, Cassandra, and graph.

3.6 Specialized Hardware for Databases – The Case for Sharing

Zsolt Istvan (ETH Zürich, CH)

In today's computing landscape, with CPU performance stagnating, there is an opportunity in using specialized hardware for accelerating databases and other data processing applications. Specialized hardware, such as FPGAs, enable a more efficient use of silicon, tailored to the problem at hand. Research efforts have already provided several promising operators/operations we could offload, but to be worth the effort hardware needs to support more than one particular operator and cater to a wide range of workloads instead. The question emerges: In the context of databases and data science applications, what is a right way of sharing specialized hardware, such as FPGAs? What if the overhead of sharing is larger than its benefits? Is it possible to retain some level of programmability even if hardware is hidden behind libraries?

While there are already ways to a virtualize the fabric of an FPGA, i.e. offer a slice of the device to each workload or tenant, these are made less attractive by high overheads both in additional circuit size and reprogramming time. Conversely, we could hide FPGAs behind the abstraction of a library that can be called by multiple applications/tenants but this method requires a decision at design time about what operations to hand off to the hardware.

In this talk I sketch the idea of a compromise: On the one hand, FPGAs should be exposed more as libraries or services, rather than bare hardware. This helps with integration and there should be enough similarity between workloads in an appliance, or data processing applications in the cloud, to be able to distill a common "core" functionality a priori. On the other hand, by looking only at the common parts we might miss additional opportunities of acceleration. Plugging custom hardware blocks into the "core" functionality should be still allowed, through use-case-specific interfaces. Since the scope of these custom blocks would be restricted their overhead would be negligible and they could be more realistically generated by high level synthesis tools, even stitched together at runtime from small building blocks (e.g., sub-operators). This, together with sharing, would enable a more flexible use of accelerators.

3.7 Latest trends in Networking

Andrew W. Moore (University of Cambridge, GB)

In this talk, I discuss networking trends at the edge of database environment. Trends in networking go beyond performance improvement – where these are not bound by physical constraints. Datacenter scaling has meant network component costs driven down and led to new configuration (P4) and operation (OpenFlow) languages and revitalisation of formalisms. Such function abstraction of SDN also means the division of work among end-system and network devices is now more flexible; many opportunities throughout.

3.8 Rethinking Memory System Design

Onur Mutlu (Carnegie Mellon University, US)

The memory system is a fundamental performance and energy bottleneck in almost all computing systems. Recent system design, application, and technology trends that require more capacity, bandwidth, efficiency, and predictability out of the memory system make it an even more important system bottleneck. At the same time, DRAM and flash technologies are experiencing difficult technology scaling challenges that make the maintenance and enhancement of their capacity, energy efficiency, and reliability significantly more costly with conventional techniques. In fact, recent reliability issues with DRAM, such as the RowHammer problem, are already threatening system security and predictability.

In this talk, we first discuss major challenges facing modern memory systems in the presence of greatly increasing demand for data and its fast analysis. We then examine some promising research and design directions to overcome these challenges. We discuss three key solution directions: 1) enabling new memory architectures, functions, interfaces, via more memory-centric system design, 2) enabling emerging non-volatile memory (NVM) technologies via hybrid and persistent memory systems, 3) enabling predictable memory systems via QoS-aware memory system design.

3.9 Discussion/Poster Talk: Toward More Automated Specialized Hardware

Pinar Tözün (IBM Almaden Center – San Jose, US)

As the hardware becomes more non-uniform and heterogeneous, it becomes essential for the data management systems to decide on the optimal design options based on the processor types they are running on. If the system is running on hardware that has a combination of different processing units (aggressive, low-power, special purpose, etc.), it should be able to automatically decide on which queries or transactions should run on which types of processors.

If the system is running on a hardware that has support for hardware transactional memory, it should be able to know the types of critical sections that would benefit from switching the synchronization primitive to transactions and the ones that should keep its existing synchronization method. If there are machine learning tasks that would benefit from running on GPUs or could be accelerated via FPGAs, the system should know when to offload these tasks to these units. Nevertheless, this requires a thorough understanding of the specific requirements of a data management system that can exploit the specific features of various hardware types. In addition, it requires interdisciplinary collaborations; especially involving people from data management, computer architecture, and compiler communities. Furthermore, to come up with more economically viable solutions in this area, we need to reach out to cloud providers so that they start building software and hardware infrastructures with heterogeneity in mind.

3.10 Managing Machine Learning on Modern Hardware

Ce Zhang (ETH Zürich, CH)

Main reference H. Zhang, J. Li, K. Kara, D. Alistarh, J. Liu, C. Zhang, "The ZipML Framework for Training Models with End-to-End Low Precision: The Cans, the Cannots, and a Little Bit of Deep Learning", in Proc. of the 34th Int'l Conf. on Machine Learning, Vol. 70, pp. 4035–4043, PMLR, 2016.

URL http://proceedings.mlr.press/v70/zhang17e.html

Recently there has been significant interest in training machine-learning models at low precision: by reducing precision, one can reduce computation and communication by one order of magnitude. We examine training at reduced precision, both from a theoretical and practical perspective, and ask: is it possible to train models at end-to-end low precision with provable guarantees? Can this lead to consistent order-of-magnitude speedups?

We present a framework called ZipML to answer these questions. For linear models, the answer is yes. We develop a simple framework based on one simple but novel strategy called double sampling. Our framework is able to execute training at low precision with no bias, guaranteeing convergence, whereas naive quantization would introduce significant bias. We validate our framework across a range of applications, and show that it enables an FPGA prototype that is up to 6.5x faster than an implementation using full 32-bit precision. We further develop a variance-optimal stochastic quantization strategy and show that it can make a significant difference in a variety of settings. When applied to linear models together with double sampling, we save up to another 1.7x in data movement compared with uniform quantization. When training deep networks with quantized models, we achieve higher accuracy than the state-of-the-art XNOR-Net.

Finally, we extend our framework through approximation to non-linear models, such as SVM. We show that, although using low-precision data induces bias, we can appropriately bound and control the bias. We find in practice 8-bit precision is often sufficient to converge to the correct solution. Interestingly, however, in practice we notice that our framework does not always outperform the naive rounding approach. We discuss this negative result in detail.

3.11 Adaptive FPGA-based Database Accelerators – Achievements, Possibilities, and Challenges

Daniel Ziener (TU Hamburg-Harburg, DE) and Jürgen Teich (Friedrich-Alexander-Universität Erlangen-Nürnberg, DE)

Joint work of Ziener, Daniel; Becher, Andreas; Dennl, Christopher; Teich, Jürgen; Meyer-Wegener, Klaus
Main reference D. Ziener, F. Bauer, A. Becher, C. Dennl, K. Meyer-Wegener, U. Schürfeld, J. Teich, J. S.Vogt, H. Weber, "FPGA-Based Dynamically Reconfigurable SQL Query Processing", ACM Transactions on Reconfigurable Technology and Systems (TRETS), Vol. 9(4), pp. 25:1–25:24, 2016.
URL http://dx.doi.org/10.1145/2845087

In this talk, we show the achievements of our research on adaptive FPGA-based database accelerators as well as possibilities and challenges of such a system. Our approach exploits the capabilities of partial dynamic reconfiguration of FPGAs [1, 2, 3, 4]. After the analysis of an incoming query, a query-specific hardware processing unit is generated on-the-fly and loaded on the FPGA for immediate query execution. With such adaptive accelerator we achieve a 10 times better performance and up to 30 times better energy efficiency compared to software only solution on an x86-based server [5]. The challenges are the needed flexibility as well as the support of many different operations for accelerate real multi-user database scenarios.

References

1 Ziener, D., Bauer, F., Becher, A., Dennl, C., Meyer-Wegener, K., Schuerfeld, U., Teich, J., Vogt, J. S., Weber, H. *FPGA-Based Dynamically Reconfigurable SQL Query Processing*. ACM Transactions on Reconfigurable Technology and Systems (TRETS) 9 (2016) 25:1–25:24

2 Dennl, C., Ziener, D., Teich, J. *On-the-fly Composition of FPGA-Based SQL Query Accelerators Using A Partially Reconfigurable Module Library*. In: Proceedings of the IEEE International Field-Programmable Custom Computing Machines Symposium (FCCM), Toronto, Canada (2012) 45–52

3 Dennl, C., Ziener, D., Teich, J. *Acceleration of SQL Restrictions and Aggregations through FPGA-based Dynamic Partial Reconfiguration*. In: Proceedings of the IEEE International Field-Programmable Custom Computing Machines Symposium (FCCM), Seattle, USA (2013)

4 Becher, A., Bauer, F., Ziener, D., Teich, J. *Energy-Aware SQL Query Acceleration through FPGA-Based Dynamic Partial Reconfiguration*. In: Proceedings of the International Conference on Field Programmable Logic and Applications (FPL). (2014) 662–669

5 Becher, A., Ziener, D., Meyer-Wegener, K., Teich, J. *A Co-Design Approach for Accelerated SQL Query Processing via FPGA-based Data Filtering*. In: Proceedings of 2015 International Conference on Field-Programmable Technology (FPT '15), IEEE (2015)

4 Working groups

4.1 Breakout Session on "Machine Learning and Databases: Options for Integration"

Michaela Blott (Xilinx – Dublin, IE)

Within this breakout session we considered a number of possibilities to integrate machine learning capabilities with databases:

1. The first idea was to offer machine learning algorithms as user defined functions (UDFs).
2. The second suggestion proposed, considered machine learning as part of the extract, transform and load (ETL) functions for data wrangling, cleansing and preparation and as part of the postprocessing. In the latter case, the data as retrieved from the database would simply feed into a ML analysis tool. A given example was HANA feeding into SAS.
3. We briefly discussed self-tuning databases leveraging machine learning algorithms, but considered these as more extravagant.

Numerous reasons for and against integration were pooled together. Pro integration were the following reasons:

1. Machine learning offers new insights and intelligence that can be gained out of existing data.
2. Secondly debugging would be easier.
3. Databses could become more useful.
4. Database community could bring relevant insights to the machine learning community.
5. Databases would provide the means to tracking and version control for machine learning data.
6. This provides an opportunity to add provenance correctly to databases.
7. Integration provides the capability to provide further acceleration.
8. Integration provides the possibility to leverage insights from 50years of database research to new types of problems.
9. There are obvious commonalities, as described in the next paragraph.

On the downsides, the following conclusions were reached:

1. Security concerns were raised.
2. We found concerns around the clash of an open-source software base with customer-owned private code bases.
3. Integration fo software stacks such as python and R might present numerous difficulties, as well as development support, profiling, and version control.
4. Concerns around database administration were raised.
5. Machine learning presents a huge spectrum of algorithms with very different compute and storage requirements.
6. Machine learning algorithms are not mature yet and undergo continuous substantial change.
7. Finally, given the huge compute times of machine learning algorithms that significantly outweighs the transfer time from database to analytics system, the overall benefit of integrating the systems might be negligible.

In summary, the consensus drifted towards keeping machine learning and databases separate. It might be an opportunity for another seminar in a years time or a separate workshop series as well as a more speculative investigation with a closer look at compute, transfer times and specific algorithms. Perhaps a subset of the overall spectrum, such as decision treed and random forests, which are lighter in compute requirements, could provide an ideal opportunity to start with.

4.2 Breakout Session "Accelerators: Where to position them in the stack?"

Zsolt Istvan (ETH Zürich, CH), Markus Dreseler (Hasso-Plattner-Institut – Potsdam, DE), Ken Eguro (Microsoft Research – Redmond, US), Babak Falsafi (EPFL – Lausanne, CH), Henning Funke (TU Dortmund, DE), Peter Hofstee (IBM Research Lab. – Austin & TU Delft), Eliezer Levy (Huawei Tel Aviv Research Center – Hod Hasharon, IL), Gilles Pokam (Intel – Santa Clara, US), Kenneth Ross (Columbia University – New York, US), Margo Seltzer (Harvard University – Cambridge, US), and Pinar Tözün (IBM Almaden Center – San Jose, US)

Accelerators for data processing are emerging and there are multiple possible hardware architectures and interaction models that have been or could be implemented. Our goal was to discuss about where future accelerators should be located in the stack, so that the DB can best take advantage of them. We concluded that accelerators should be part of the platform and managed by the operating system (OS) but expose a programming interface (API) tailored to the application's need, in our case the database engine. This requires some level of co-design between the accelerator and the applications on top.

We defined what we mean by an accelerator for the purposes of this discussion: "A piece of specialized hardware, that is non-critical, but improves efficiency". The consensus in our group was that the accelerator has to be part of the platform and needs to be managed by the operating system. Operating systems already take care of configuring and multiplexing access to different hardware devices that constitute the platform. Any future accelerator should expose to the OS its multiplexing rules, e.g. level of parallelism, whether it is run-to-completion, etc. And in turn the OS should be able to configure the accelerator for context switching, and deploy tasks on it for the applications.

The layers above the platform and OS, however, should be able to provide task descriptors that are specific to their domain (these could be in declarative or imperative form) and specify what data to use for input and where to write outputs. Using co-designed APIs that act as a bridge between accelerator and database ensures that acceleration functions can take advantage of information about workloads etc. readily available in the database. The actual API will be dependent on the implementation of an application/database and such details as the location of the data, whether the accelerator acts synchronously or asynchronously, etc. In some cases it could be beneficial to provide multiple APIs to the same accelerator depending on the level of abstraction the application on top can best utilize.

4.3 Breakout Session "Future Memory/Storage for Databases"

Zsolt Istvan (ETH Zürich, CH), Goetz Graefe (Google – Madison, US), Peter Hofstee (IBM Research Lab. – Austin & TU Delft), Stefan Manegold (CWI – Amsterdam, NL), Ingo Müller (ETH Zürich, CH), Kenneth Ross (Columbia University – New York, US), Kai-Uwe Sattler (TU Ilmenau, DE), Eric Sedlar (Oracle Labs – Redwood Shores, US), Margo Seltzer (Harvard University – Cambridge, US), and Thomas Willhalm (Intel Deutschland GmbH – Feldkirchen, DE)

With future storage and memory solutions becoming increasingly programmable and the line between the two becoming more blurred, the challenge of integrating these with databases for better performance emerges. We are also presented, however, with the opportunity of shaping the functionality of future storage solutions to match the needs of the database. During our discussions we approached the question from both the angle of a computer architect (what features to add or expose) and a database person (how to take advantage of existing/future features). We identified co-design as being necessary and the output of our discussion is a list of meta-data the database can provide to the storage devices (see attachment).

Given that databases have already good understanding about the data they manage, and also about their own internal data structures, they should be able in the future to provide hints about access patterns, etc. to the operating system (OS) to guide the choice of memory/storage device. Furthermore, in case the devices support offloading functionality, this information could be directly shared with the devices. We explored what information about data the DB could push down to the OS and devices (e.g., whether data is persistent or transient, compressible or not, represents pointers or user data, what is the expected read-write ratio, etc.) and ranked these by expected impact. Access pattern information was identified as the most useful that DBs could provide to smarter storage devices of the future. The expectation is that when data is streaming, for instance, bandwidth could be exploited better by the device, and in case it is pointer-driven access, the devices could perform smarter prefetching, or even traverse structures on their own.

From a database perspective we formulated a requirement to guide the design of offloading functionality in storage/memory, namely that stability and predictability is more important than best-case speedup. For instance, a stable 2x improvement will be preferred over a non-guaranteed 5x, because this would make integration with the query planner and cost functions more feasible.

We also concluded that co-design is preferred, because it is important that the DB receives back from the storage status information specific to the meta-data mentioned before. Trying to hide the acceleration features behind opaque layers in the OS will mean missed acceleration opportunities.

4.4 Reproducible Floating-Point Aggregation

Ingo Müller (ETH Zürich, CH), Gustavo Alonso (ETH Zürich, CH), Andrea Arteaga, and Torsten Hoefler

Industry-grade database systems are generally expected to produce the same result for queries run on the same input as software vendors and cloud providers may be liable for non-reproducible behavior of their systems. However, the numerous sources of indeterminism in modern systems make exactly reproducible results difficult to achieve. This is particularly true if floating-point numbers are involved, where the order of the operations affects the final result.

As part of a larger effort to extend database engines with data representations more suitable for machine learning and scientific applications, in this paper we explore the problem of making relational GroupBy over floating-point formats bit-reproducible, i.e., we make any execution of the operator produce the same result up to every single bit. We start from recent algorithms from high-performance computing that make the summation of floating-point numbers into a single sum bit-reproducible. In a SQL query with GroupBy, however, each of the potentially many groups produces a separate sum, so consecutive inputs do not necessarily aggregate to the same sum. As a consequence, a GroupBy operator with a naïvely integrated bit-reproducible summation algorithm incurs a slowdown of factor 4 to 12 compared to the same operator using conventional summation. We thus explore how to modify existing GroupBy algorithms to make them bit-reproducible and efficient. We are able to reduce the slowdown due to reproducibility to a factor between 1.9 to 2.4, thereby providing a solid basis for supporting more complex operations directly in relational engines. We show the trade-offs offered by the different algorithms in extensive experiments and give recommendations on how to use them in practice.

4.5 Breakout Session "Memory and Storage"

Jens Teubner (TU Dortmund, DE), Alexander Böhm (SAP SE – Walldorf, DE), Sebastian Breß (DFKI – Berlin, DE), Markus Dreseler (Hasso-Plattner-Institut – Potsdam, DE), Babak Falsafi (EPFL – Lausanne, CH), Henning Funke (TU Dortmund, DE), Onur Mutlu (Carnegie Mellon University, US), Gilles Pokam (Intel – Santa Clara, US), Jürgen Teich (Friedrich-Alexander-Universität Erlangen-Nürnberg, DE), Pinar Tözün (IBM Almaden Center – San Jose, US), Annett Ungethüm (TU Dresden, DE), and Daniel Ziener (TU Hamburg-Harburg, DE)

A number of exciting new technologies around memory and storage have emerged in the recent years, including Hybrid Memory Cubes (HMC) or various technologies that provided non-volatile memory at near-RAM speeds. In a breakout session, members of the hard- and software communities discussed the opportunities that the new technologies may open up for database acceleration.

The breakout group concluded that the new technologies could indeed help solve the memory bandwidth problems that main-memory database engines have kept struggling with. Most importantly, scatter/gather-type data accesses would excellently fit many database workloads. On commodity hardware, however, such accesses are known to perform poorly. The modern memory technologies have ample bandwidth inside the memory chip, but are limited by the link to the rest of the system. With little add-ons to the memory chips, chips could re-arrange data following a database's needs before shipping data to the CPU. Suitably equipped and configured, a memory chip could, for instance, convert between row- and column-oriented views of a database table on demand. More advanced applications of the concept could even offer "near-memory computing" with feature-rich processing capabilities right inside the memory chips.

The obvious part where non-volatile memories will affect database engines is transaction management. It seems yet unclear, however, how that could look like, since the technology will blur the line between data structures (e.g., database indexes) designed for on-disk or for in-memory.

5 Open problems

5.1 Data management for HPC clusters

Spyros Blanas (Ohio State University – Columbus, US)

Processing petabyte-sized datasets quickly will inevitably require datacenter-scale computers. In such computers, "hot" storage consists of petabytes of DRAM that is fragmented across tens of thousands of nodes. Nodes are interconnected in unique topologies through proprietary networking hardware. The "cold" data access path is a parallel file system (such as Lustre) with many petabytes of network-attached storage.

The optimizations a DBMS currently performs are insufficient when data management becomes a datacenter-scale challenge. We posit that this unique hardware platform is more than a disaggregated collection of compute, memory and storage resources. We instead envision a data processing system that optimizes query processing holistically for the datacenter and carefully orchestrates data processing tasks for better performance.

We identify the following research opportunities: query optimization that predicts and ameliorates detrimental I/O interference; a distributed buffer pool that proactively places data in a multi-tiered but fragmented storage hierarchy; query execution algorithms that directly interface with high-end, low-latency interconnects; holistic I/O optimization for processing massive arrays by automatically collecting semantically richer metadata to guide data placement.

5.2 Tradeoffs in Energy Efficient Data Management

Henning Funke (TU Dortmund, DE), Stefan Noll, and Jens Teubner (TU Dortmund, DE)

Database systems frequently use an over-proportional amount of energy to offer high performance. This has the disadvantage that high energy consumption limits the ability to scale out due to thermal design constraints. One approach to reduce the power intake is to adjust the power states of hardware. This can be done with frequency scaling of CPUs and DRAM and allows finding a better trade off between energy and performance when adapting power states to workload characteristics. However, in many cases users do not want to give up performance. Therefore, we propose to look for other trade offs than performance for energy. One direction, we consider is trading in accuracy for performance.

5.3 Persistent Memory: Opportunities and Challenges

Thomas Willhalm (Intel Deutschland GmbH – Feldkirchen, DE)

Intel and Micron are bringing a new type of non-volatile memory "3D XPoint" to the market. It promises to close the gap between DRAM and flash, in that it will be denser and less expensive than DRAM but faster than flash. This will not only allow to create very fast SSDs but NV-DIMMs based on 3D XPoint. These DIMMs will therefore enable the implementation of persistent memory where durable data can be accessed directly with load/store semantics.

Such a solution will not only leap-frog the performance of storage, but also tear down the barriers of block I/O. On the other hand, it poses new challenges on correctness and a more complex memory hierarchy. Last but not least, the question of data replication over a comparably slow network will become a new challenge.

Participants

- Anastasia Ailamaki
EPFL – Lausanne, CH
- Gustavo Alonso
ETH Zürich, CH
- Carsten Binnig
Brown University – Providence, US
- Spyros Blanas
Ohio State University – Columbus, US
- Michaela Blott
Xilinx – Dublin, IE
- Alexander Böhm
SAP SE – Walldorf, DE
- Peter A. Boncz
CWI – Amsterdam, NL
- Sebastian Breß
DFKI – Berlin, DE
- Markus Dreseler
Hasso-Plattner-Institut – Potsdam, DE
- Ken Eguro
Microsoft Research – Redmond, US
- Babak Falsafi
EPFL – Lausanne, CH
- Henning Funke
TU Dortmund, DE
- Goetz Graefe
Google – Madison, US
- Christoph Hagleitner
IBM Research Zurich, CH
- Peter Hofstee
IBM Research Lab. – Austin, US & TU Delft, NL
- Stratos Idreos
Harvard University – Cambridge, US
- Zsolt Istvan
ETH Zürich, CH
- Viktor Leis
TU München, DE
- Eliezer Levy
Huawei Tel Aviv Research Center – Hod Hasharon, IL
- Stefan Manegold
CWI – Amsterdam, NL
- Andrew W. Moore
University of Cambridge, GB
- Ingo Müller
ETH Zürich, CH
- Onur Mutlu
Carnegie Mellon University, US
- Thomas Neumann
TU München, DE
- Gilles Pokam
Intel – Santa Clara, US
- Kenneth Ross
Columbia University – New York, US
- Kai-Uwe Sattler
TU Ilmenau, DE
- Eric Sedlar
Oracle Labs – Redwood Shores, US
- Margo Seltzer
Harvard University – Cambridge, US
- Jürgen Teich
Friedrich-Alexander-Universität Erlangen-Nürnberg, DE
- Jens Teubner
TU Dortmund, DE
- Pinar Tözün
IBM Almaden Center – San Jose, US
- Annett Ungethüm
TU Dresden, DE
- Stratis D. Viglas
Google – Madison, US
- Thomas Willhalm
Intel Deutschland GmbH – Feldkirchen, DE
- Ce Zhang
ETH Zürich, CH
- Daniel Ziener
TU Hamburg-Harburg, DE