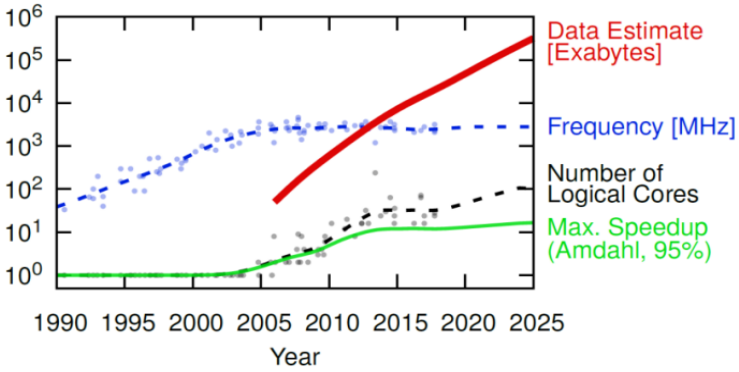
Research Statement

Zsolt István

In my research I investigate how specialized hardware can be used to lift computation and communication bottlenecks in data management and data processing systems. I work in the intersection of databases, systems and specialized hardware (FPGAs), combining ideas from these areas with the goal of designing the next generation of data processing systems in a holistic way. To achieve this, I plan to collaborate in the future with other scientists to look across layers, from hardware through software frameworks to the algorithms used. Such a holistic approach is required because the quantity of information that needs to be stored and processed in today's datacenter grows faster than the performance of general purpose processors. For decades the latter has been increasing conforming to Moore's law, but today this trend has leveled off. As a result, data-intensive applications suffer from the widening gap between data and compute, and in order to change the status quo we need on the one hand, to look beyond traditional processor- and rack-scale designs, and on the other hand, to design software systems that can take full advantage of hardware features.

The graphic on the right illustrates the gap between data growth and processor speeds. Even though it is possible to add more transistors to a CPU, using these to create additional cores is unlikely to benefit applications (the figure shows that even if 95% of the application's execution can be parallelized perfectly, speedup on multi-cores over a single thread is capped below 20x). As an alternative to adding conventional cores, specialized processing elements can be used to extend the capabilities of CPUs. This way computation can be handled more efficiently and we can narrow the gap between data growth and compute capacity.



Based on a plot layout by K. Rupp. Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten; 2010-2015 by K. Rupp. Data growth estimate by C. Maxfield.

Another challenge that I address in my research comes from the fact that most data processing applications are designed to scale out to tens or hundreds of machines. They are often organized into compute and storage tiers and even though this allows them to tackle large problem sizes and to use resources more elastically, it introduces various data movement inefficiencies across the system. As a result, it is increasingly important to make sure that data is moved only when and where necessary. One way of achieving this is by pushing parts of the computation closer to the data source (offloading), and thanks to the changing hardware landscape, this can be done using specialized hardware in an efficient manner in different layers of the architecture.

Even merely five years ago, though we could already demonstrate improvements through the use of hardware accelerators, the adoption of heterogeneous architectures in the datacenter faced a push-back. Today, however, it has become clear that we need to look beyond traditional CPUs to face the computational requirements of emerging workloads. We see more and more programmable network-interface cards, GPUs, and even field programmable gate arrays (FPGAs) appearing in the datacenter, for instance at Microsoft, Amazon, Baidu and Huawei. FPGAs are reprogrammable chips that can implement any logic, but once programmed, act as specialized integrated circuits. This makes it possible to express algorithms in ways that are fundamentally different from CPUs or GPUs, which means that FPGAs are particularly promising in overcoming the above mentioned challenges in the datacenter. Overall, this is an exciting time to be working in the field of systems because the widespread adoption of heterogeneous hardware opens up great opportunities in accelerating large-scale applications.

1. Current Research

Distributed Smart Storage

With the prevalence of scale-out applications, distributed storage in the form of general purpose key-value stores (e.g., Redis, Amazon S3) or as application-specific data stores (e.g., in Oracle Exadata), occupies more and more resources in the datacenter. The design of nodes providing this service is, however, often un-balanced. If regular server machines are used, they can provide rich functionality and complex computation since there are many conventional cores available, but are quite inefficient because of the over-provisioning of compute capacity and the bandwidth mismatch between storage media, CPU, and network. Networked storage devices, on the other hand, are better balanced but at the price of offering very limited options for offloading data processing.

In my dissertation, I show that it is possible to have a more balanced design, with the right amount of compute resources to match both network and storage bandwidth. We achieved this in **Caribou** [1] through **network/application co-design using an FPGA**. We designed the internal data structures and pipelines in such a way to handle virtually any mix of random-access operations that can be received over a 10Gbps network link. As a result, Caribou's performance reaches over 11 million requests/s on regular TCP/IP connections [3], which is almost 3x higher than what can be achieved with software over the same commodity network. Furthermore, in order to ensure that data is not lost in case of failures, Caribou nodes implement a **full-fledged replication protocol directly in hardware** [4] that introduces overhead only in the order of microseconds.

To mitigate the data movement bottleneck between storage nodes and the rest of the application, I explored **different ways data can be filtered inside the storage node while matching the streaming read bandwidth** of the underlying storage. One challenge has been to implement functionality in a way that ensures throughput that is not dependent on the complexity of the filtering expression, so that offloading can never slow down data retrieval, even if none of the items can be filtered out.

Caribou is open source and can be used as an "idea accelerator" for prototyping near-data processing solutions and to explore integration with different applications. Thanks to its modular design, new functionality can be easily added, leading to faster prototyping of systems that take advantage of processing in the storage layer.

Database Acceleration

While decades of optimizations ensure that transactional workloads and traditional operators perform well, analytical workloads continue to evolve and the growing data sizes put pressure on databases. Working with colleagues from the Systems group, I investigated how specialized hardware can turn compute-bound operations into bandwidth-bound ones through careful re-design of algorithms.

- In **DoppioDB** [2], a hybrid CPU-FPGA database engine, I worked on a **regular expression matching operator** aimed at text queries. These queries are expensive in software and often replaced by index-based workarounds to ensure low response times. In contrast to software, our hardware solution offers constant throughput (several GB/s) even for complex expressions. This not only makes queries containing regular expressions cheaper, but also predictable in terms of runtime. Although regular expressions on FPGAs have already been studied in a rich body of related work they mostly target matching on pre-defined expressions and therefore were unsuitable for our use-case. We developed a circuit that allows for run-time parameterization, and this way can provide software-like flexibility while retaining hardware-like performance for this operator.
- In **IBEX** [5], a disk controller for MySQL that supports SQL offloading, I worked on **aggregating tuples in-storage for a group-by predicate** before they would be sent to the CPU. In this project one of the main challenges was to find the sweet spot in hybrid computation. In the case of group-by aggregation, for instance, we implemented a hash-table based approach that, in order to guarantee constant rate processing that matches the disk's bandwidth, forwarded tuples with hash-conflicts to software instead of stalling the pipeline for collision resolution. In the typical case, when all groups fit on the FPGA, an order of magnitude speedup can be achieved. Even in the case when there are a high number of groups (tens of thousands) and not all fit on the FPGA, the overall performance of the system is always equal to or better than the software-only version.

2. Planned Research Activities

Accelerating Tomorrow's Workloads

Most data science applications run on tens or even hundreds of machines. Some of them, for instance training jobs for machine learning based on a parameter server approach, can even scale to a thousand machines. At this scale it is important to ensure that all resources are used efficiently. This includes the processing inside a node, and more importantly, the network links between nodes. In terms of compute resources, machine learning and AI workloads are the main driving force behind the increase in innovation in domain-specific computation, with Google's Tensor Processing Unit (TPU) and NVIDIA's Volta GPU being just two examples of datacenters entering a "hybrid era". At the same time, there are untapped opportunities for acceleration at the communication level, at various layers in the architecture. My interest is to work together with scientists from other fields to **identify common patterns and operations in machine learning workloads that could be pushed down to hardware**. In my dissertation, I showed how it is possible to build distributed systems with specialized hardware that utilize the available network bandwidth to the fullest, with little impact from the mix of operations in a workload. This applies both to client-facing operations, such as the data retrieval and filtering in Caribou nodes [1], or the consensus protocol running between hardware nodes [4] used to implement storage-side replication for fault tolerance. These insights together with input from domain specialists could be used to build a new generation of programmable NICs and network-attached devices to reduce data movement bottlenecks.

To make sure that data science applications can continue scaling in the future, accelerating software that exists today might not be enough. Instead, in some cases, we will have to **design future data processing systems using a holistic approach**. By being able to optimize across different layers in the software and hardware stack, it is possible to carry out complex analysis more efficiently. Especially in scenarios where the raw data comes from sources such as sensors or IoT devices with high velocity and it has to be cleaned and transformed before use, we can gain in efficiency by co-designing what we traditionally think of as storage and processing layers. Working using a holistic approach requires tight collaboration of scientists at different levels of the application, but will enable gaining better insights using overall fewer resources.

Databases on Tomorrow's Hardware

Big Data applications, such as machine learning and graph analytics, work on input data that is not organized by the relational model, but is represented instead as wide vectors, vertex/edge descriptors or images with meta-data. A large portion of this information, however, originates from relational databases in the first place and has to be copied out for processing. If databases could extend their operator offering this overhead could be reduced and we could retain features such as ACID transactions, fault tolerance and a declarative query language (SQL) for manipulating the data. Furthermore, being able to apply the results of these operations directly in the database could also be beneficial. In order to extend the capabilities of databases with compute-intensive operators, however, accelerators such as GPUs and FPGAs have to be integrated with the query execution engine. Otherwise using a database would be prohibitively slower than the dedicated processing systems.

In order to successfully integrate accelerators with databases, we need to address at least two challenges:

- First, we have to **design hybrid operators** that combine CPUs and specialized hardware, such as FPGAs, to solve compute-intensive problems utilizing each of the processing elements to their full potential. Somewhat counter-intuitively, the best way to partition the work is not necessarily based on traditional operator boundaries, but instead it can be beneficial to either fuse multiple operators or to explore offloading at a smaller granularity inside a single operator (e.g., the partitioning step in a hash-join algorithm). To ensure that each new operator idea doesn't require re-evaluating all design options, I plan to work on systematic ways of determining the right division of work in such heterogeneous operators.
- Second, we need to **extend the query optimizer and cost functions** to encompass emerging accelerators and heterogeneous platforms. Even though the runtimes of hardware operators are usually more predictable than their software counterparts, collocating different types of processing on the same FPGA and scheduling them is an open problem and raises challenging research questions. However, solving these will enable us to extend databases with novel compute-intensive operators and to support workloads usually carried out by separate systems.

System Support for Specialized Hardware

The diversity of data processing and acceleration features in the network and in end-hosts raises an additional challenge: integration with the software stack and applications. Both the applications will have to be aware of the acceleration features and hardware devices must become shareable between different workloads and adaptable at run-time to the particular needs of the applications. I plan to work on **mechanisms for managing acceleration functionality** at the operating system level in the end-hosts and at the (SDN) controller level for the network.

Furthermore, I want to explore how accelerators could be designed such that they can be used by multiple workloads, even at the same time. This entails **developing hardware circuits that are flexible and can switch between workloads** without overhead, or suspend a task and resume it later. This is a departure from the traditional view of FPGAs as dedicated accelerators and raises many research challenges, but in the long term shareable hardware accelerators should make the widespread adoption of hybrid architectures viable.

3. Collaboration with Industry

The field of systems research is evolving at an unprecedented rate, and much of the novel work is being carried out at companies. For instance, when I was an intern at Microsoft Research in 2014 we used the first generation Catapult boards, and today, only a few years later, the next generation boards are already widely deployed in the Azure Cloud to offload virtual networking tasks. These devices now open up opportunities in near-data processing and FPGA-based distributed systems at unprecedented scale.

During my studies at the Systems Group I have had the opportunity to work on cutting-edge research platforms and collaborate with researchers from industry on multiple occasions. Having early access, for instance, to the Intel Xeon+FPGA platform was a great enabler for our research on hybrid database engines. My internship at Xilinx Labs in Dublin and our continued collaboration was also valuable experience. In the future I plan to continue my existing collaborations and start new ones with industry partners in an effort to transfer ideas and lessons from academia to industry.

Selected Articles

[1] Caribou: Intelligent Distributed Storage. Z. István, D. Sidler, G. Alonso. Proceedings of the 43rd Int'l Conference on Very Large Data Bases (VLDB'17), 2017.

[2] Accelerating Pattern Matching Queries in Hybrid CPU-FPGA Architectures. D. Sidler, Z. István, M. Ewaida, G. Alonso. ACM SIGMOD/PODS Conference (SIGMOD'17), 2017.

[3] Low-Latency TCP/IP Stack for Data Center Applications. D. Sidler, Z. István, G. Alonso. 26th Int'l Conference on Field Programmable Logic and Applications (FPL'16), 2016.

[4] Consensus in a Box: Inexpensive Coordination in Hardware. Z. István, D. Sidler, G. Alonso, M. Vukolic. 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI '16), 2016.

[5] IBEX: An Intelligent Storage Engine with Support for Advanced SQL Offloading. L. Woods, Z. István, G. Alonso. Proceedings of the 40th Int'l Conference on Very Large Data Bases (VLDB'14), 2014.

[6] Histograms as a Side Effect of Data Movement for Big Data. Z. István, L. Woods, G. Alonso. ACM SIGMOD/PODS Conference (SIGMOD'14), 2014.