Rack-scale Data Processing System

J. Giceva, D. Makreshanski, C. Barthels, A. Dovis, G. Alonso
Systems Group, Department of Computer Science, ETH Zurich
{name.surname}@inf.ethz.ch

1. INTRODUCTION

Rack-scale computers (RaSC) are becoming the building blocks of modern data centers. They are characterized by new hardware technologies that promise terrabytes of main memory, thousands of cores connected with low-latency, high-bandwidth interconnects, including RDMA over new network fabrics such as silicon photonics. At the same time, modern workloads involve a large volumes of frequently updated data. On such workloads enterprises often have requirements for data freshness and SLA performance guarantees.

An important class of modern data processing systems are database appliances. They focus on optimizing all layers of a system to achieve the best performance for a given application (Oracle's *Exadata*, SAP's *BWA*, etc.). These optimizations include multiple levels of co-design and tight integration of different layers of the system stack (hardware/software, operating system/database, etc.) often resulting in performance levels almost unattainable with general purpose systems built using commodity components.

We want to investigate some of the research questions for rack-scale computers from an application's perspective by building a rack-scale database appliance (SwissBox [1]) that addresses modern workload characteristics and requirements.

2. PROJECT VISION

Our goal is to build a data appliance for RaSC that leverages the benefits of cross-layer optimization and provides support for elasticity, flexibility and resiliance in noisy environments.

To achieve that, we separate the storage from the data processing layer (Figure 1). This has already been successfully applied in some commercial database appliance solutions (Exadata, BWA). The two layers communicate over a scalable interconnect fabric using different network interfaces and technologies. Both layers are internally distributed (possible implemention using the programming abstractions presented in FaRM [2]), and are composed of heterogeneous building components. For example, the processing layer can have a DBMS that uses the storage layer, but could also run machine learning or graph processing algorithms.

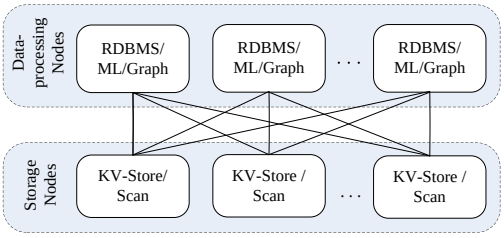

Figure 1: Architecture

3. INITIAL SYSTEM PROTOTYPE

The *data storage layer* of our prototype is composed of both key-value (KV) stores and clock scans (Crescando [4]). The main components of the *data processing layer* is SharedDB [3], which implements high level operators (join, sort, group-by aggregation, etc.) in a way it can answer thousands of queries belonging to a given workload in predictable time with low latency tail. The two layers communicate via RDMA over InfiniBand.

While building the prototype, our goal is to revisit the design and implementation of the two layers based on the available technologies, hardware properties, workload characteristics and user requirements. For example, how to use RDMA over InfiniBand for different data transfer patterns. Some will include a lot of small point look-ups in the KV stores, others require large bulks of data to be transferred, or even data to be streamed between the storage and processing nodes. One approach is to re-design the storage layer software to better suit RDMA-type data transfers. An alternative is to identify the most common data transfer patterns required by modern workloads and find a suitable match of technologies to achieve that. There are other interesting challenges, such as how to manage data transfers over the network as a resource for multiple concurrent requests, baring in mind concrete user performance requirements. Lastly, we evaluate the performance of the system and identify possible opportunities for future optimization steps.

4. CONCLUDING REMARKS

The main goal of our talk is to provide another perspective, application and workload-driven, for designing system software on RaSC. By revisiting the design paradigms used for data appliances and by characterizing the modern enterprise workloads and requirements, we present our design and early prototype for rack-scale data processing system.

We believe it is a great opportunity for interdisciplinary collaboration across many layers of the stack, including both at the software and the hardware components.

5. REFERENCES

[1] G. Alonso, D. Kossmann, and T. Roscoe. Swissbox: An architecture for data processing appliances. In *CIDR*, pages 32–37, 2011.

[2] A. Dragojević, D. Narayanan, M. Castro, and O. Hodson. FaRM: Fast Remote Memory. In *NSDI 14*, pages 401–414, 2014.

[3] G. Giannikis, G. Alonso, and D. Kossmann. SharedDB: killing one thousand queries with one stone. *PVLDB*, 5(6):526–537, Feb. 2012.

[4] P. Unterbrunner, G. Giannikis, G. Alonso, D. Fauser, and D. Kossmann. Predictable performance for unpredictable workloads. *PVLDB*, 2(1):706–717, Aug. 2009.