Near Data Processing within Column-Oriented DBMSs for High Performance Analysis

Tobias Vincon[1] und Ilia Petrov[2]

Abstract: Rapidly growing data volumes push today's analytical systems close to the feasible processing limit. Furthermore, performing near-time precise analytics and knowledge extraction becomes trend in social media, intelligent cars or business analysis. Evaluation involves complex analyses of larger, more detailed datasets, which consumes more time than enterprises can invest.
Massive parallelism, is one possible solution to reduce the computational time of analytical algorithms, by utilising many-core CPUs or by distributing the entire system over multiple compute nodes. However, data transfer becomes a significant bottleneck since it blocks system resources moving *data-to-code*. Technological advances allow to economically place compute units close to storage and perform data processing operations close to data, minimising data transfers and increasing scalability. Hence the principle of Near Data Processing (NDP) and the shift towards *code-to-data*. In the present paper we claim that the development of NDP-system architctures becomes an inevitable task in the future. Especially, analytical DBMS like HPE Vertica have multiple points of impact with major advantages which are presented within this paper.

1 Introduction

The aggregate worldwide data volume grows exponentially. Mobile phones, social media and intelligent cars persist more data than ever nowadays with rapidly increasing size. Built-in sensors and cameras have tremendously higher resolutions and recording speeds will further enlarge the data volume to around 35ZByte in 2020 (CSC [Co12]). New kinds of DBMSs were developed to handle this enormous amount of information more efficiently [Fo11]. They exhibit architecture, data structures and access patterns suitable for analytic algorithms. In addition, significant effort is put into the research of in-memory solutions by database vendors to provide near-time or real-time evaluations. Nevertheless, even present-day data volumes can only be processed with highly scalable databases within distributed data centres involving latest hardware.

While data volumes are growing exponentially, improvements in algorithmic complexity remain painfully slow. In other words, increasing data volumes yield larger N, yet with typical algorithmic complexities of $O(N \cdot log(N))$, $O(N^2)$ or even much worse, reasonable response times become less feasible.

Consider the following example: a small hosting company is managing 6000 hosted systems. To ensure optimal SLA compliance and resource utilisation all of them need to

[1] Hewlett Packard Enterprise, Böblingen, Germany - Data Management Lab, Reutlingen University, Germany, tobias.vincon@hpe.com

[2] Data Management Lab, Reutlingen University, Germany, ilia.petrov@reutlingen-university.de

be considered, yet due to high analytics complexity and large data volumes the administrators target merly 600 of them. The systems monitoring infrastructure logs 25 parameters per system such as disk, CPU or network utilisation. The average size of a log entry is between 64 bytes and 4K on average 500 bytes. The reporting infrastructure can provide average data samples (per parameter per system) in different frequencies: sample per year, month, working day, day, hour, minute down to second. Hence $N \in (200, 1\,600, 96\,000, 5, 760, 000)$. The yearly data volume is 39 TB (sample/second) or 1TB (sample/min). Computing the *co-variance* is a simple way to estimate the similarity of one parameter on two systems; its computational complexity is $O(N)$. For Z systems; the complexity increases to $O(\frac{Z^2 \cdot N}{2})$. Assume 600 systems, 25 parameters/system and minute averages ($Z = 600 \cdot 25 = 15000, N = 96000$) and it would require a machine with 10 Tflops (floating point operations per second). This requires approx. 400 CPU cores to compute the co-variance matrix within a second. Assuming a storage system that can read the raw per minute data at 2 GB/s it will take approx 8.5 minutes to transport the data to the CPUs. *In Big Data environments both computational complexity and data transfers play a significant role.*

Many modern systems tackle this issue with two methods: (a) massive parallelism; and (b) scalability. Ideally, these are balanced systems (Amdahl's balanced systems law) exhibiting shared-nothing architectures. In essence, this implies partitioning N data entries into P partitions and distributing data onto P Nodes, while introducing some degree redundancy R (so now each node contains $P \cdot R$ partitions) and performing computation with unchanged complexity in parallel on each node.

Unfortunately, these solutions suffer the dark side of data transfers. Whenever data is transported from an I/O device to the computing unit time elapse, energy is wasted and, even worse, computational resources are blocked. Consequently, under the *data-to-code* principle, system resources cannot be efficiently utilised. For the above example this means each node performs $O(\frac{Z^2 \cdot N/P}{2})$. Yet, since data cannot be perfectly distributed for all workloads, the data transfers still amount to $R \cdot P$ on each node (when no inter-node data transfers take place). Assuming 10 nodes and $R = 2$, for above example this results in $1TB * R/10$ or approx. 200GB/node. Hence each node will complete computation within a second, yet the data transfer will take 10 seconds assuming 2GB/s storage.

Modern technological advances allow to economically place compute units close to storage and perform data processing operations close to data, minimising data transfers and increasing scalability. For instance, high-performance enterprise storage systems collocate Flash storage and FPGA or GPU compute units[Ba14, HYM15]. Prototypes demonstrate a performance increase of 30x or more, if certain data processing operations are decoupled, pushed down and evaluated close or within the storage. Such architectures combine the *code-to-data* principle with *Near Data Processing (NDP)*. Although NDP approaches were proposed in the 70s, only now the production of such devices became economical and represents an emerging trend. Efficiently utilising such requires a complete redesign of DBMS data structures and interactions.

This paper purposes NDP's scopes of applications, especially for the column-oriented DBMS HPE Vertica as one popular representative of analytical databases. Besides this, NDP might also have significant impact on the performance of further compute-intensive system types e.g. scientific databases, key-value stores and statistical packages. Current work in the NDP space is presented and own points of impact on column-stores (HPE Vertica) are identified. The emerging visions and their implementation can be used for future evaluations.

2 Related Work

Beginning in the early 70s, researchers tried to place processing units close to storage devices since they realised that data transfer consumes time and resources, leading to a bottleneck of data-intensive workloads. In 1969, William Kautz already presented his CLIM array[Ka69] which automatically keeps its persisted data in order. One year later, Harold Stone's logic-in-memory computer[St70] appears to have processing capabilities for systems while it is only a logic-in-memory array performing high-speed buffering of the CPU. Producing these memory arrays with computing units was uneconomically till today. Great progress in the research of 3D stacking [HYM15] and FPGA enable to produce memory chips associating logical components with low costs. Realising these technology, 3D-stacked nanostores are going to integrate computing with non-volatile memory and network interfaces e.g. Hewlett Packard Enterprise's Memristor [Ra11].
Sparking the interest of researchers again, a complete workshop was dedicated to the NDP topic in 2014 including work about reasons of its revival and current research results of companies [Ba14]. In 2016, another collection of papers about NDP's areas of application are published defining several services [Fa16]. E.g. the machine learning system DaDianNao used for visual recognition within neural networks, the active memory cube leveraging 3D memory stacks by transforming stored data through customised interpretation or the memristive dot-product engine enabling matrix vector multiplication on memory arrays. Achieving 16x performance and energy advantages, Mingyu Gao et. al present hardware and software for an NDP architecture able to perform MapReduce, graph processing and deep neural networks analytical use cases [GAK15]. Another concrete application for databases is JAFAR [Xi15], a NDP accelerator which is able to push selects directly to the memory in modern column-stores achieving a 9x performance increase.
However, the implementation of NDP capabilities within databases is considered sparely. Especially within column-stores like HPE Vertica which are best-of-breed for analytical use cases the effects might be tremendous. Points with strong impact are presented in the following Section.

3 NDP's Points of Impact within Analytical DBMSs

Most modern DBMS are similarly structured including a query processor, transaction manager, buffer manager and disk storage. Reusable data structures like hash or bitmap indices and B^+-trees are commonly utilised to efficiently support the database. However, both, components and common data structures of DBMSs must be rethought to benefit from NDP. This includes pushing database operations down to NDP storage to reduce the data transfer dramatically [Xi15]. The following list categorise the most decisive points NDP can support the DBMSs, in particular the column-oriented database HPE Vertica.

[1. Data Storage] Data storage designs are located most closely to the actual I/O device. Including page formates and data structures like linked lists or directories, their definition has to be known by the NDP's processing module to efficiently access data. Special attention must be paid to the locality of data since the compute units only operate on persisted data of the associated memory dies. Efficient data distribution over several NDP nodes leverages complex calculations and increases parallelism. *Compression* and encoding are implemented techniques to reduce space consumption. Within HPE Vertica these are highly important to shrink the increased redundancy caused by its multiple possible projections. Therefore, NDP must be able to operate on, send, and receive compressed data. Encoding should only be applied to data transfers if and only if the encoded data is less space consuming. Concepts of bandwidth consumption might support the development as their goal is the data transfer reduction. Principles like buffer replacement policies will probably have less impact. Solely, a cache for intermediate results or frequent executed operations must be managed.
[2. Query Evaluation] Evaluating and executing entire or parts of queries is highly probable to be handled by NDP's computing units. Selections, for example, can be pushed down to pre-fetch data for later analytical calculations, as JAFAR does. Further operations could be implemented similarly supporting most of the relational operators of a database. Within HPE Vertica these might be the selection, mask, projection, aggregation and bitstring operations since they reduce data transfer. Remaining operations like decompress, concat, permute and join may produce result sets larger than the input data and are therefore suitable under controlled conditions. Internal operations of HPE Vertica's Tuple Mover can be optimised by NDP. *Mergeout* and *Moveout* are restructuring the internal data containers or move data from its ROS to WOS[3]. However, in any case special architectural designs of the database like the partitioning, segmentation and late materialisation of HPE Vertica must be respected.
[3. Security] Securing data is often realised among other things with encryption. Since encryption algorithms are characterised by high CPU workloads, this is another improvement candidate of NDP.

4 Conclusions

NDP advantages impact data-centric systems considerably. By reducing data transfer, operations of data-intensive applications like databases accelerate and their parallelism is increased significantly. Especially column-oriented DBMS, best-suited for analytical use cases, take advantage of this principle. Giving the list of NDP's points of impacts, this paper presents how HPE Vertica's performance can be further increased in the future.

References

[Ba14] Balasubramonian, R.; Chang, J.; Manning, T.; Moreno, J. H.; Murphy, R.; Nair, R.; Swanson, S.: Near-Data Processing: Insights from a MICRO-46 Workshop. IEEE Micro, 34(4):36–42, July 2014.

[Co12] Cooperation, Computer Science: Infographic Big Data. Technical report, 2012.

[3] HPE Vertica utilises a Read Optimised Store (ROS) and Write Optimised Store (WOS) to increase the write performance. Its Tuple Mover act as broker optimising containers and moving data

[Fa16] Falsafi, B.; Stan, M.; Skadron, K.; Jayasena, N.; Chen, Y.; Tao, J.; Nair, R.; Moreno, J.; Muralimanohar, N.; Sankaralingam, K.; Estan, C.: Near-Memory Data Services. IEEE Micro, 36(1):6–13, Jan 2016.

[Fo11] Forum, Leading Edge: Data rEVOLUTION. Technical report, 2011.

[GAK15] Gao, M.; Ayers, G.; Kozyrakis, C.: Practical Near-Data Processing for In-Memory Analytics Frameworks. In: 2015 International Conference PACT. pp. 113–124, Oct 2015.

[HYM15] Hassan, Syed Minhaj; Yalamanchili, Sudhakar; Mukhopadhyay, Saibal: Near Data Processing: Impact and Optimization of 3D Memory System Architecture on the Uncore. In: Proceedings of the 2015 International Symposium on Memory Systems. MEMSYS '15, pp. 11–21, 2015.

[Ka69] Kautz, W. H.: Cellular Logic-in-Memory Arrays. IEEE Transactions on Computers, C-18(8):719–727, Aug 1969.

[Ra11] Ranganathan, P.: From Microprocessors to Nanostores: Rethinking Data-Centric Systems. Computer, 44(1):39–48, Jan 2011.

[St70] Stone, H. S.: A Logic-in-Memory Computer. IEEE Transactions on Computers, C-19(1):73–78, Jan 1970.

[Xi15] Xi, Sam Likun; Babarinsa, Oreoluwa; Athanassoulis, Manos; Idreos, Stratos: Beyond the Wall: Near-Data Processing for Databases. In: Proceedings of the 11th International Workshop on Data Management on New Hardware. DaMoN'15, pp. 2:1–2:10, 2015.