UNIVERSITY OF AMSTERDAM

Multi-Core Parallelism in a Column-Store

M.M. Gawade

Abstract

While the amount of data generated and collected keeps on growing rapidly, the added value is not the data itself but comes from finding patterns and extracting information and knowledge from the data. Analytical data management systems are the primary tools to facilitate efficient analysis of huge data volumes.

The computing power of a single machine keeps improving at an unabated rate. However, the increase of computing power no longer stems from increased clock-speed, but rather from extensive parallelization inside the CPU including multi- and many-core architectures. Consequently, (re-)designing existing and new software, including data management systems, to efficiently and effectively using the entire available computing power of rapidly changing and ever more complex hardware architectures has become an important science and engineering challenge.

While analytical database query processing in principle lends itself well to parallelization, finding a good (let alone optimal) way of parallelizing an arbitrary given query depends on numerous parameters including the kind and complexity of the query itself, detailed characteristics of the ever more complex hardware architectures, characteristics of the data and concurrent workloads (some of which, in particular the latter two, might not be known upfront), and is known to be a computationally hard problem.

The research reported in this thesis addresses several challenges of improving the efficiency and effectiveness of parallel processing of analytical database queries on modern multi- and many-core systems, using an open-source column-oriented analytical database management system, MonetDB, for validation. In contrast to the existing work we also broaden the research from focusing on individual operators and algorithms to consider the entire system and process holistically.

A prime prerequisite to achieving resource-efficient parallel query execution is a detailed understanding of the impact of the various parameters sketched above. Recognizing limitations of existing techniques and tools, we design and develop new visual analysis techniques and tools that help to identify and rank performance bottlenecks of parallel query execution on multi-core systems.

Deploying these tools in multiple showcases revealed that in particular as the number of CPU cores grow rapidly with multi- (let alone many-) core CPUs, finding an optimal degree of parallelization becomes increasingly difficult. Static parallelization techniques easily fail by using too low degree of parallelism, and thus leaving resources (cores) unused, or using too high degree of parallelism, and thus suffering from synchronization and other overheads.

This observation inspired us to design and develop a novel learning based adaptive technique for multi-core parallel plan generation using query execution feedback. This techniques proves to be particularly efficient with concurrent workloads, a scenario which is very common in practice but has been largely uncharted in database query parallelization research.

To further increase the compute power of a single machine, multi-socket systems (accommodating multiple multi-core CPUs) have become a commodity. However, while providing transparent access from each CPU to the entire available memory, access performance is non-uniform, i.e., each CPU has faster access to ”its own” local part of the memory, but slower access to the ”remote” memory of other CPUs. Ignoring this non-uniformity in memory access in parallel database query evaluation leads to non-optimal performance and undesired performance variations.

We show that using a simple technique where a multi-socket system is treated as a distributed shared nothing database system, the remote memory accesses could be constrained thereby having a controlled query execution performance.

Many-core system architectures are the latest trend to imitate GPU style parallel execution where there are 240 threads on 60 cores in a Xeon-Phi processor. However, data transfer on the PCIe bus which connects Xeon-Phi co-processor to the host, is a bottleneck due to the limited bandwidth. We analyzed the effect of streaming execution of selected queries, to utilize PCIe bandwidth optimally, to understand possibility of Xeon-Phi knights corner architecture usability in data analytical workloads.

This thesis contributes to our understanding of the multi- and many-core CPU landscape in the context of analytical database systems, exemplified by the MonetDB columnar system. Many of the lessons, experiences and insights gained are valuable for the emerging analytical database systems.