Query processing for datacenter-scale computers

Spyros Blanas

The Ohio State University
blanas.2@osu.edu

Introduction

Quickly exploring massive datasets requires an efficient data processing platform. Parallel database management systems were originally designed to scale to a handful of nodes, where each node keeps recent ("hot") data in memory and has directly-attached hard disk storage for infrequently accessed ("cold") data. To keep up with the growing data volumes, the research focus is shifting towards rack-scale architectures. Rack-scale database management systems, including Oracle Exadata, the IBM PureData System and the Microsoft Analytics Platform System, combine powerful nodes with directly-attached storage for "warm" data with hundreds of terabytes of network-attached storage for "cold" data.

Processing even larger datasets quickly will inevitably require datacenter-scale computers. Although details of the hardware configurations of commercial datacenters are scarce, one can use scientific computers for data-intensive applications as a proxy. In these datacenters, "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 query processing 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 query processing kernel that optimizes query processing holistically for the datacenter and carefully orchestrates data processing tasks for better performance. We identify the following research directions towards realizing this vision.

Research opportunities

Query optimization that predicts and ameliorates detrimental I/O interference: The I/O cost of a query plan is commonly modeled as a linear function of the number of disk seeks and the transferred data volume. An implicit assumption is that all I/O requests will be processed near the peak throughput rate. The parallel file system of a datacenter-scale computer, however, is concurrently used by thousands of nodes. As multiple I/O requests target the same storage component, the aggregate I/O pattern becomes increasingly random. This leads to highly variable I/O costs due to queuing. For query optimization to be effective, the I/O cost model needs to assess the performance impact of I/O interference and then extrapolate to imminent I/O activity. In addition, query plans that operate on "cold" data need to be robust to high variance in I/O latency. Finally, a DBMS needs to account for coordination-based I/O optimizations that coalesce I/O into larger sequential requests.

A distributed buffer pool that proactively places data in a deep and fragmented storage hierarchy: The binary classification between cached ("hot") data and "cold" data fails to capture the depth of the data storage hierarchy in datacenter-scale computers. The storage hierarchy typically includes components such as: node-local memory; rack-local memory; memory outside the rack; rack-local non-volatile storage; and I/O burst buffers between the datacenter and the parallel file system. In a datacenter-scale computer, the buffer pool needs to become aware of the relative *distance* to the data. The distance metric needs to clearly capture both intra-node and inter-node effects. Intra-node effects reflect the cost to access data in the local cache hierarchy or in other NUMA nodes. Inter-node effects reflect the access path to the data in the network fabric; it includes the impact of the network topology on latency and the impact of link congestion on throughput. Furthermore, lineage information can transform the buffer pool from a passive responder to an active participant that proactively places input data and intermediate results closer to the point of data consumption.

Query execution algorithms that directly interface with high-end, low-latency interconnects: Prior work has already shown that a query processing engine that uses TCP/IP for communication does not fully utilize a fast network. High-end computers take network performance one step further through proprietary interconnects that can offer lower latency and higher throughput than commodity server networking (such as EDR InfiniBand). These proprietary interconnects can perform scatter/gather operations and synchronization directly in the network. In addition, global address space operations can be offloaded to the hardware, instead of emulating address translation and cache coherence in software. To better utilize these unique interconnects, a query processing engine needs to interface with the networking hardware at a lower level than through the InfiniBand verbs abstraction.

8th Biennial Conference on Innovative Data Systems Research (CIDR '17). January 8–11, 2017, Chaminade, CA, USA

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235