Diamond: Diagnostics and Monitoring for the Data Center

Zaheer Chothia, Desislava Dimitrova, John Liagouris, and Timothy Roscoe
Systems Group, Department of Computer Science, ETH Zurich

Problem Statement

Modern data centers, of both the enterprise and cloud flavour, tolerate downtime as good as none. At the same time, data center diagnostics face the challenge to construct an up-to-date, homogeneous view of the data center state, based on a vast amount of heterogeneous data which are monitored and processed by disparate software and hardware. Shortcomings of diagnostics are not acceptable since they have direct consequences on the quality of service delivery. The heterogeneity of diagnostic tasks and correspondingly varied monitoring data adds a new dimension to the "one size does not fit all" problem. Along with specialized data management systems to address the distinct requirements of each task, we need to effectively coordinate their operation within a unified framework.

Technical Challenges

The large amount of data available for data center diagnostics is both a blessing and a curse. The more we know about the data center, the better we are in its efficient and reliable management, conditioned on our ability to properly interpret, correlate and analyse large sets of disparate data. This is challenging due to several aspects. First, for a consolidated view, a diagnostic system needs to analyze various data formats produced by the many components of the data center. Second, depending on the diagnostics, different types of processing are appropriate. Rapid detection of anomalies or unexpected behaviour requires real-time processing over streaming data, while trends analysis and predictive models do well with batch-style analysis of integrated fresh and historic data. Third, the sheer amount of data (several hundred TBs per day) challenges its effective presentation. Data center operators are best supported by visualizations that allow them to quickly gain insights into the data center and easily change observation levels, e.g., application vs. network level. The isolation of relevant data at each level should come at minimum processing overhead.

Motivation

Some of the previous challenges may seem similar to those addressed in Data Warehouses (DW) research through the Extract-Trasform-Load (ETL) procedures; however, our goal here is to manipulate the data in-situ and not to integrate the different data sources into a new, separate knowledge base. The reason behind this decision is threefold: (i) ETL processes cannot be easily migrated into new data ecosystems without considerable manual work, (ii) organizational boundaries often pose restrictions in sharing data amongst different departments, and (iii) there are tremendous efficiency gains from reusing specialized tools already in deployment. Furthermore, in such a setting, we have various types of OLTP and OLAP workloads which cannot be easily handled by a single of-the-shelf system. These workloads include complex graph processing (e.g., for topology-aware service deployment), relational operations (e.g., `GROUP-BY` queries for reconstructing user sessions from application message logs), stream processing of rich data with various attributes all of which are necessary for a cross-layer analysis. Based on our preliminary efforts, none of the existing MapReduce-style systems, Graph Databases, Streaming Engines, and RDBMSs fulfills all previous requirements.

Difficulties Faced

We have made an early attempt building a proof of concept on top of several open-source tools from three main areas: (i) RDBMSs, (ii) Graph DBs, and (iii) Streaming Engines. Although our approach is still at its infancy, the results from applying it to a real data center (from a large enterprise in the travel industry) showed that it can already provide valuable insights to the data center operators. The difficulties we faced during this process can be summarized in the following: whilst it is possible to build the envisioned system on top of existing platforms, e.g., Naiad, we observed that it involves significant manual work and complex engineering decisions. The main source of difficulty is that the underlying systems adopt rather different programming models which are hard to integrate (e.g., graph-based models vs. streaming data models), and this can be only achieved by relying on the low level APIs of each system and build almost everything from scratch. A second deficiency is that, although all systems offer high-level languages, the latter are not very useful for expressing variations of the built-in algorithms; for instance, variations of a shortest path algorithm depending on the particular data center configurations.

Road to Solution

A first step to tackle the aforementioned problems is to define an appropriate model of the data center. The model must be flexible enough to handle all different features of the data (e.g., loose schema, incompleteness, temporal and spatial dimensions, dependencies, etc.), and should also allow a uniform access pattern to all platforms and data sources. Clearly, such a model is of no value without a high-level language for coordinating the underlying systems and for manipulating the data in a transparent way. Following the architectural paradigm of the state-of-the-art big data platforms, the desired language should be declarative, providing the administrator with the necessary abstraction to easily define analytic tasks that are internally performed by heterogeneous systems.

The cross-layer platform we strive to construct can reinforce a wide range of management and diagnostic procedures such as troubleshooting of application-rooted network failures or preemptive detection of network hotspots and congestion points, to name a few. We are convinced in the platforms potential to meet the evolving needs of data center managers and our current efforts are dedicated to its full-scale realisation.