

Polygraph - An Arrow Format for Large Apache Calcite Query Plans and Multi-Language Data Systems (Lightning Talk)

Adam Kennedy¹

¹Voltron Data Inc., 650 Castro Street Suite 120, PMB 96571, San Francisco, CA 94041

Abstract

The maturity and substantial investment in Apache Calcite establish it as the open source standard for query planning and optimization across numerous data tools. Nevertheless, utilizing Apache Calcite for dynamic query planning in a diverse tool stack with multiple languages has proven challenging. Through the integration of Apache Arrow, we introduce Polygraph: a language-independent, parse-free, and efficient format for query plans. Its purpose is to enhance plan interoperability, diminish latency and overheads, and facilitate dynamic query optimization. This experimental format allows for the efficient exchange of query plans between tools in diverse languages with minimal serialization overhead.

While future query engines are steering away from Java, Calcite remains the solitary mature option for query planning across a broad spectrum of workloads. Few alternatives come close to matching its features. However, Calcite relies on tree-based JSON or XML plan representations that do not readily lend themselves to certain optimizations and necessitate substantial overhead for serialization, I/O, and parsing. The commingling of planners and engines across languages is rare, unusual, and complex. Such approaches typically result in ad hoc, internal formats with limited reusability. Addressing these challenges, Polygraph relocates the query plan to Arrow. Polygraph employs a graph structure encoded with columnar storage techniques. Preliminary experiments indicate an order of magnitude reduction in query plan size compared to JSON encoding, without incurring copying and serialization overheads. Arrow provides zero-copy, shared-memory, and parse-free capabilities, along with fast RPC via Arrow Flight. In this representation, plan consumers only need to load the components and properties of a query plan required for a given computation. These efficiencies substantially reduce the latency between plan generation and query execution. Moreover, we envision significant potential for other advancements, including resource planning, ML preprocessing, and integration into ML training and inference.

Until recently, there was no urgent imperative to represent query plans efficiently. However, the escalating complexity and size of query graphs will persist as data tools become more deeply integrated into intricate ML workloads. Polygraph's agile and decomposable graph representation empowers data engines to contribute to query optimization and resource management. Enhanced integration with top-tier ML systems becomes more viable, facilitating the incorporation of run-time compute planning and resource management into the query plan, utilizing tools like Apache Acero. The benefits extend beyond improvements in space efficiency and latency. Query sub-plans can be optimized in-situ using real-time hardware metrics. Value relations and broadcast tables can be seamlessly embedded in the plan as Arrow objects, accessed in a zero-copy manner. Large models can be directly incorporated into the query plan, incurring no loading cost until required. Increased investment in query plan representations, exemplified by Polygraph, supports the data community in keeping pace with advancements in new architectures and problem domains, such as AI.

Keywords

Query planning, Apache Calcite, Apache Arrow


Joint Workshops at 49th International Conference on Very Large Data Bases (VLDBW'23) – Second International Workshop on Composable Data Management Systems (CDMS'23), August 28 - September 1, 2023, Vancouver, Canada

✉ adam@voltrondata.com (A. Kennedy)

🌐 <https://voltrondata.com/> (A. Kennedy)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)