Distritbuted Stream Processing With Query Compilation

Supervision: *Prof. Dr. Tilmann Rabl, Dr. Predro Silva*

Students: *Ivan Ilic, Youri Kaminsky, Till Lehmann, Tobias Niedling*

As the value of new data generally decreases over time, analyzing it as early as possible becomes increasingly important. This is where stream processing frameworks come into play. In contrast to traditional batch processing, stream processing engines (SPEs) continuously process data as soon as it arrives from an unbounded data source (*stream*). Thereby stream processing queries have no finite runtime and steadily generate results.

To cope with ever-growing amounts of data, state-of-the-art stream processing engines like Apache Flink [1] choose to scale-out and distribute the workload across multiple nodes. Thus, they achieve higher performance and very good scalability. On the downside, however, recent analyses expose their deficiency in sufficiently utilizing modern hardware capabilities (i.e. multi-core processors), as they often rely on the JVM as an overhead introducing abstraction layer [2].

Latest research explores increasingly sophisticated approaches to enhance query performance on single processing units. Compiling queries into strongly optimized, hardware-conscious executables has proven to surpass traditional approaches in terms of throughput and latency by orders of magnitude [3] [4]. The benefit applies especially to long-running streaming queries, where the constant compilation overhead becomes negligible.

Although distribution as well as query compilation have been proven to increase the maximum throughput of SPEs, no research exists that analyses how these distinct optimization techniques perform in combination. Thus, to get the best of both worlds, it is necessary to evaluate how well the latest research achievement of compiled queries can be harnessed and integrated into the industry-standard approach of distributing streaming workloads on multi-node clusters.

In this work, we present a novel streaming engine prototype that generates compiled queries for streaming data and distributes the workload across multiple nodes. Our experimental evaluation shows that our proposed approach achieves $12.6 \times$ higher throughput than Apache Flink and scales super-linearly when distributing across multiple nodes on a modern 16-nodes high-performance cluster.

Our contributions are summarized as follows:

- We demonstrate the performance optimization potential that can be realized with hardware-aware executables by implementing a hardcoded C++ query, that is later used as an evaluation baseline.
- We add another engine to our set of baselines by implementing an iterator-style C++ streaming engine that is similar in concept to Apache Flink.
- We implement a novel streaming engine prototype that combines a) distribution over multiple nodes with b) query compilation for highly optimized stream processing.
- We implement a multi-threaded data generator capable of streaming data at specified rates and use it to evaluate our distributed, query-compiled engine thoroughly.
- We perform an extensive performance evaluation of our engine prototype that includes (amongst others) comparing its performance to Apache Flink and analysing its behaviour when scaling across an increasing number of nodes.

The prototype is available on GitHub [5], where also the project report [7] and the slides of the final presentation [6] are located.

References

[1] The Apache Software Foundation, „Apache Flink,“ 2021. [Online]. Available: https://flink.apache.org/.

[2] S. Zeuch, B. D. Monte, J. Karimov, C. Lutz, M. Renz, J. Traub, S. Breß, T. Rabl und V. Markl, „Analyzing Efficient Stream Processing on Modern Hardware,“ in *Proc. VLDB Endow.*, 2019.

[3] T. Neumann, „Efficiently Compiling Efficient Query Plans for Modern Hardware,“ 2011.

[4] P. M. Grulich, S. Breß, S. Zeuch, J. Traub, J. v. Bleichert, Z. Chen, T. Rabl und V. Markl, „Grizzly: Efficient Stream Processing Through Adaptive Query Compilation,“ in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA, 2020.

[5] GitHub-Repository, „mp-ddsp-ws20,“ [Online]. Available: https://github.com/hpides/mp-ddsp-ws20.

[6] I. Ilic, Y. Kaminsky, T. Lehmann und T. Niedling, „Distributed Stream Processing With Query Compilation,“ 2021. [Online]. Available: https://github.com/hpides/mp-ddsp-ws20/blob/master/final_presentation.pdf.

[7] I. Ilic, Y. Kaminsky, T. Lehmann und T. Niedling, „Distritbuted Stream Processing With Query Compilation,“ 2021. [Online]. Available: https://github.com/hpides/mp-ddsp-ws20/blob/master/final_report.pdf.