Performance Improvements of an Event Index Distributed System

Extended Abstract

Álvaro Fernández Casaní[*]
Instituto de Física Corpuscular (IFIC),
Universidad de Valencia and CSIC
Burjassot, Valencia, Spain
Alvaro.Fernandez@ific.uv.es

Juan M. Orduña[†]
Departamento de Informática,
Universidad de Valencia
Burjassot, Valencia, Spain
Juan.Orduna@uv.es

Santiago González de la Hoz[‡]
Instituto de Física Corpuscular (IFIC),
Universidad de Valencia and CSIC
Burjassot, Valencia, Spain
Santiago.Gonzalez@ific.uv.es

ABSTRACT

Modern scientific collaborations, like the ATLAS experiment at CERN, produce large amounts of data that need cataloging to meet multiple use cases and search criteria. Challenges arise in indexing and collecting billions of events, or particle collisions, from hundred of grid sites worldwide. In addition we face challenges in the organization of the data storage layer of the catalog, that should be capable of handling mixed OLTP (high-volume transaction processing updates) and OLAP (real-time analytical queries) use cases. In order to overcome the challenge on the distributed data collection of events, we have designed and implemented a distributed producer/consumer architecture, based on an Object Store as a shared storage, and with dynamic data selection. Producers run at hundreds of grid sites worldwide indexing millions of files summing up Petabytes of data, and store a small quantity of metadata per event in an ObjectStore. Then a reference to the data is sent to a supervisor, that signals consumers to retrieve the data at the desired granularity and consolidate at a central Hadoop based data backend. In the area of the internal organization of the data, we propose an architecture based on a NoSQL backend storage, and new data schemas to better accommodate related data (reprocessings), avoiding duplicate information, and improving navigation. We propose applying memory caching techniques to improve access times for recent loaded data, which is usually the most accessed data by the end-users use cases.

CCS CONCEPTS

• **Computer systems organization → Distributed architectures**; *Grid computing*; • **Information systems** → *Data management systems*;

[*]PhD Student. 3 years in PhD program.
[†]PhD supervisor.
[‡]PhD supervisor.

KEYWORDS

Grid Computing, Hadoop, Object-Based storage

ACM Reference Format:
Álvaro Fernández Casaní, Juan M. Orduña, and Santiago González de la Hoz. 2018. Performance Improvements of an Event Index Distributed System: Extended Abstract. In *Proceedings of 47th International Conference on Parallel Processing (ICCP2018).* ACM, New York, NY, USA, Article 4, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The ATLAS experiment at CERN [1] is producing, during its Run2 phase (2015-2018), in the order of 10^{10} events or particle collisions every year. This data is stored and distributedly reprocessed at different sites worldwide using grid technologies, to extract higher level information and store it in formats more suitable to different uses. A catalog of data (all events in all processing stages) is therefore needed to meet use cases like (I) locate individual events (event picking) depending on constraints, (II) make consistency checks, including detection of duplicates and overlaps, and (III) make analytic studies over large amounts of data. The EventIndex project [2, 3] is a metadata catalogue at event level which tries to exploit technologies such as Hadoop [6]. A small quantity of metadata per event is indexed, including identifiers (run/event numbers, trigger stream, luminosity block), the trigger pattern that made the event to be recorded, and references (pointers) to the events at each processing step in all permanent files on storage.

We developed a producer/consumer architecture for the distributed data collection task of the EventIndex project with a messaging implementation [5], that has been indexing petabytes of input data, and has produced 150 TB of events meta data that are stored at the Hadoop infrastructure at CERN. This system has efficiently handled more than 10^9 messages, but during high production campaings, we detected head of line blockings on the messaging brokers. Messaging systems are designed to handle a large number of small messages, but our typical payload consists on large data files that have to be divided into smaller messages. This segmentation and re-assembly procedure is complex, and it has an effect on the brokers and consumers performance, and the scalability of the system. Since during the following runs starting in 2021 the production rates will be increased, we needed to explore other collection mechanisms.

2 IMPROVED DISTRIBUTED DATA COLLECTION

In order to solve the complexity and scalability problems, we have proposed, designed and implemented a new distributed producer / consumer architecture, based on Object Store (OBS) as a shared storage, and with dynamic data selection [4]. The producer now puts all the event index data in a OBS, without dividing the payload in various messages, and when it is done it sends a control and statistic message to the statistics queue of the broker. The control message is received by the supervisor, an entity previously called validator, now implemented with the intelligence needed to orchestrate all the procedure. This is needed because we change the model of consuming the data, from a push model where all the data is directly transmited to consumers by means of the brokers, to a pull model where the consumers are informed by the supervisor about the data they have to retrieve from the OBS. Supervisor is in charge of selecting the valid produced information and signaling consumers to retrieve the appropriate data from the OBS system. The communication for these entities is done with control and statistic messages similar to the ones from the messaging scenario, so we still use queues from the brokers to distribute the processing messages among different consumers. The supervisor entity also takes into account the possibility of some fraction of the event processing not reaching its final state, as it was done with the validator in the Messaging scenario. Now the difference is that this partial data is not being continuosly pushed to the consumers. When reaching a desired processing granularity (for example indexing all the data from a dataset), the supervisor signals the consumer with a control message which contains all the info needed to retrieve the data by the latter. This allows a consumer to consolidate information in a single step, writing a unique file in HDFS filesystem, instead of having multiple files written by several consumers like in the messaging scenario.

3 RESULTS OBTAINED

We have presented a new pull-model approach for the distributed data collection of the ATLAS EventIndex project, based on an Object Store as a shared storage and with dynamic data selection. It must be noted two key differences in this new approach. First, the entire payload from a given producer can be potentially stored within a single object regardless of its size, avoiding complex issues like message groups and transactions. This avoids blockings due to this matter, and so better workload distribution and scalability adding new consumers when necessary. This was not effectively possible with the messaging approach and in addition this allows us to use different data encodings and compression, reducing the amount of conveyed data. Second, the behavioural change from a push-model to a pull-model. This model allows to use the OBS as a temporary storage, eliminating the need to consume duplicated produced data. Only valid data is retrieved from the OBS and consolidated into bigger, more suitable files in the Hadoop HDFS filesystem. It reduces the amount of data that is consumed, and so the network usage from the OBS to the final HDFS backend. We can also avoid extra and expensive cleaning tasks on the Hadoop cluster. The reduction of complexity and the resource usage, and better performance of the distributed data collection, has improved the experience for final users, that have seen reduced the Traversal time, or latency, of the datasets indexed data. Overall the results show that the new approach can efficiently support large-scale data collection for big data environments, like the next runs of the ATLAS experiment at CERN. The ObjectStore based solution is now the reference implementation for the ATLAS EventIndex project, and is being currently used in production since 2018.

4 REMAINING OBJECTIVES

During recent years the growth of main memory capacity has enabled the development of in-memory big data management and processing systems [7]. The second objective of this thesis is to design and implement a data storage layer exploring these systems and capable to handle OLTP and OLAP mixed workloads, to satisfy the described use cases, and which improve the usability and performance. Exploring solutions that involve memory caching for recent ingested data would improve many of our use cases. In addition, we can analize several patterns that are currently in use in our Hadoop storage backend that are subject to improvement. Currently many of the reprocessed and related information is stored in different files, duplicating many information due to some restrictions. A small fraction of the data is duplicated to HBase to allow random fast access for the event-picking use case. We are aiming to avoid duplicate information, and provide a unique and coherent dataset for all use cases and workloads. One of the first milestones is to study and analyze storage schemas to try to better accommodate related data (reprocessings), avoiding duplicate information, and improving navigation. Also one of the possibilities related with this is to study storing data related to 'triggers' to columnar storage, as an approach to improve the use cases that involve analitics over this kind of data.

REFERENCES

[1] ATLAS Collaboration. 2008. The ATLAS Experiment at the CERN Large Hadron Collider. *Journal of Instrumentation* 3, 08 (2008), S08003.

[2] D. Barberis, S.E. Cárdenas Zárate, J. Cranshaw, A. Favareto, A. Fernández Casaní, E.J. Gallas, C. Glasman, S. González De La Hoz, J. Hřivnáč, D. Malon, F. Prokoshin, J. Salt Cairols, J. Sánchez, R. Többicke, and R. Yuan. 2015. The ATLAS EventIndex: Architecture, design choices, deployment and first operation experience. *Journal of Physics: Conference Series* 664, 4 (2015), 042003.

[3] D. Barberis, J. Cranshaw, A. Favareto, A. Fernández Casaní, E. Gallas, S. González de la Hoz, J. Hřivnáč, D. Malon, M. Nowak, F. Prokoshin, J. Salt, J. Sánchez Martínez, R. Többicke, and R. Yuan. 2016. The ATLAS EventIndex: Full chain deployment and first operation. *Nuclear and Particle Physics Proceedings* 273-275 (2016), 913–918.

[4] A Fernandez Casani, D Barberis, A Favareto, C Garcia Montoro, S GonzÃąlez de la Hoz, J Hřivnáč, F Prokoshin, J Salt, J Sanchez, Többicke, R Yuan, and ATLAS Collaboration. 2017. ATLAS EventIndex general dataflow and monitoring infrastructure. *Journal of Physics: Conference Series* 898, 6 (2017), 062010. http://stacks.iop.org/1742-6596/898/i=6/a=062010

[5] J Sánchez, A Fernández Casaní, and S González de la Hoz. 2015. Distributed Data Collection for the ATLAS EventIndex. *Journal of Physics: Conference Series* 664, 4 (2015), 042046.

[6] Tom White. 2012. *Hadoop: The Definitive Guide.* O'Reilly Media, Inc.

[7] H. Zhang, G. Chen, B. C. Ooi, K. L. Tan, and M. Zhang. 2015. In-Memory Big Data Management and Processing: A Survey. *IEEE Transactions on Knowledge and Data Engineering* 27, 7 (July 2015), 1920–1948. https://doi.org/10.1109/TKDE.2015.2427795