Data pipelines in the cloud: elastic execution with dynamic parallelism

M2 internships in company/research laboratory co-supervision

- Start Date: February / March 2023
- Duration: 6 months
- Location: SAP France (Levallois-Perret) and LIP6 (Paris)
- Funding: about 1400 euro/month
- Co-supervisors: Hubert Naacke (hubert.naacke@lip6.fr), Bernd Amann (bernd.amann@lip6.fr) and Eric Simon (eric.simon@sap.com)

Context and Motivation

Nowadays, institutions and companies manage their data with a wide variety of applications which were not designed to communicate with each other. On the other hand, there is a very strong need to design new data management and analysis services that will add value to the data that is there. Since it is practically impossible to migrate all applications and their data into an integrated system, the current solution is to build analytic data pipelines to facilitate the data flow between operations that perform complex processing, including collecting data from multiple sources, transforming it, generating AI models through learning, and storing it in multiple destinations. In practice, a data pipeline can contain hundreds of operations, and it can evolve repeatedly by being populated with new operations or new data. Thus, with the increasing number of pipelines to be designed and deployed, it is crucial to dispose of high level data pipeline definition languages, tools to deploy and control the execution of data pipelines and efficient solutions to optimize the execution of complex operations on large volumes of data.

In this context, SAP has developed the SAP Data Intelligence (DI) software for the automatic configuration and deployment of data pipelines. These pipelines use a flow-based programming model [3]. Each pipeline operation corresponds to a program (Python, node.JS, ...) or a call to an external API (e.g., Spark job) that is deployed using an adapted Docker [2] image/container. Kubernetes services provide deployment and orchestration of these images on hyperscaler platforms like AWS, Google Cloud, Azure etc.

A performance problem arises at large scale when a pipeline contains long operations processing massive data. A first solution was designed in the context of an SAP/LIP6 internship to parallelize operators [4]. In this solution, the way to consume/produce data is described using data sorting and partitioning functions. This allows the data to be partitioned and distributed to process operators in parallel. The principle of the method is to first define the properties of a "divide and conquer" mapping in the JSON configuration of an operator. These properties allow to automatically transform a DI pipeline into a new parallelized DI pipeline with several replicas (identical copies) of the initial operator, each running in parallel on different parts of the operator's input data. A "dispatch" operator is injected into the data pipeline to split the input data stream into different partitions and a "collect" operator is injected to aggregate the output of the replicas into a single output. The replicas are aggregated into a single output data stream.

The first experiments show that this parallelization solution allows to improve the performance of data pipelines, but does not allow to obtain optimal performance in real environments, which need to estimate and to dynamically adapt the operator replication/data parallelization degree in relation to the volumes of data exchanged, the calculations performed and the available resources.

Objective and challenges

The objective of this internship is to propose new methods to facilitate and optimize the deployment and execution of parallelized data pipelines. This raises several scientific and technical challenges:

- Estimating the replication degree: How many replicas should be deployed for each operation to be processed in parallel? To answer this question, we need to estimate the benefit of parallel processing as a function of the number of replicas, the amount of data to be processed and the CPU consumption of an operation. This benefit must also be related to the cost of using the machines running data pipelines in the cloud, in order to determine an optimal number of replicas for a certain budget.
- Elastic deployment: How can we adapt the number of replicas to dynamic changes in available resources and associated costs? This demands for new solutions to allow the number of replicas (degree of parallelism) of an operator to be dynamically changed without interrupting the pipeline.

Internship goals and tasks

Internship #1. The goal of the first internship is to evaluate the performance of the parallelization method on different types of stateful operators by varying the CPU load of the operator, the size of the operators state, and the size of the messages dispatched to the replicas. The evaluation will be run on a Kubernetes cluster deployed on a hyperscaler platform. Through this evaluation, we expect to learn the configuration parameters that provide the greatest parallelization benefit and some suggestions for improving the parallelization method.
Tasks:

- Propose a model to estimate the overhead incurred by adding operations that partition data and distribute it to replicas in the pipeline.
- Design a method to observe the execution of the pipeline and detect an overload (underload) situation.
- Determine the new degree of parallelism that will improve pipeline performance.

Internship #2. The goal of the second internship is to implement dynamic dispatch and collect operators which automatically adapt to the scaling up or down of the number of replicas of a parallelized operator. For the dispatch operator, the strategy must guarantee that no message is lost in case of scaling down. For the collect operator, the strategy must guarantee that all messages produced by the replicas are properly collected and possibly re-ordered in case of scaling up.
Tasks:

- Design a technical solution to dynamically change the number of running operator replicas and adapting the dispatch and collect operators.
- Conduct experiments using data pipeline examples to check the validity of the implemented strategies and measure their possible overhead.

The solutions will be deployed in the SAP DI environment. Comparative experiments will be implemented on the Spark parallel computing platform. For this, a solution will be designed to transform the pipeline description (written with Data Intelligence syntax) into a Spark pipeline [1] (pyspark syntax).

Expected Skills

The candidate should have *excellent experience in algorithmic and programming* (Python, Java) and *advanced knowledge of optimization and parallelization techniques* (query optimization, data parallelism, map-reduce, ...) and some *technical knowledge* of Docker/Kubernetes is also helpful.

To apply, you should send to the three co-supervisors (see email above), a CV and the grades of the last three semesters of study.

Host teams

- SAP France (Levallois-Perret)
- Equipe Bases de Données du LIP6 (Paris): `http://www-bd.lip6.fr/`

References

[1] Michael Armbrust, Reynold S. Xin, Cheng Lian, Yin Huai, Davies Liu, Joseph K. Bradley, Xiangrui Meng, Tomer Kaftan, Michael J. Franklin, Ali Ghodsi, and Matei Zaharia. Spark SQL: relational data processing in spark. In *ACM SIGMOD International Conference on Management of Data*, pages 1383–1394, 2015.

[2] David Bernstein. Containers and cloud: From lxc to docker to kubernetes. *IEEE Cloud Computing*, 1(3):81–84, 2014.

[3] Tanmaya Mahapatra. High-level graphical programming for big data applications. Master's thesis, Technische Universität München (TUM), 2019.

[4] Ludgy Vestris. Scaling up stateful and order preserving operators in DI data pipelines. Master's thesis, CNAM, SAP - LIP6, 2022.