International Journal of Aquatic Science
ISSN: 2008-8019
Vol 12, Issue 02, 2021

Technical Review Of Apache Flink For Big Data

G. Paul Davidson[1], Dr. D. Ravindran[2]

[1]*Assistant Professor, Department of Computer Applications, Bishop Heber College,Tirchy-620017*
[2]*Associate Professor, Department of Computer Science, St.Joseph College, Tirchy-620002*

Abstract: Enormous amount of data are produced day by day that leads to the development of Big Data. The main characteristics of big data are categorized as high volume, high velocity, high variety, and high veracity. These Big data's has to be analyzed properly to produces an enhanced result that will be useful or needed for the organization/business for better decision making. The information got from the big data is considered as intelligence that can be utilize by the business/organization while taking decisions and these data will provide a better operational efficiency. The data has to be processed with a help of software application. The developers build a platform that provides the need known as framework. There are so many frameworks are available as an open source for processing the big data to get the desired intelligence. In this paper, we are analyzing the various technical features of Apache Flink. Apache Flink is one of the popular open source frameworks that are used for both batch and stream data processing.

Keywords: Big Data, Apache Flink, Apache Hadoop, Apache Spark and Apache Storm.

1. INTRODUCTION

In day to day life each and every organizations are producing data in enormous. The big volumes of data are considered as big Data. These data that are producing daily may be structured and unstructured. These data has to be analyzed for insights that lead to better decisions and strategic for the business development. By using the Big Data the organizations can create new growth and opportunities in a new categories do that they can combine and analyze the data. The big data characteristic is categorized in Various V's[4] based on Volume (it will be increase in terms of terabytes, petabytes, and so on), Velocity (high speed of data movement at a rapid rate), Variety (the data may be structured, unstructured or semi-structured and it may be video, image, text or audio, geo-spatial etc), Veracity (it involves the accuracy of the data or the truthfulness in the data). The processed data can improve the customer service, identification of risk at early stages, and it improves the operational efficiency. The big data has to be managed properly for the functions like storage, sorting, processing and analysis the volume of data that cannot be handled by the existing database system. So Frameworks are used provides a foundation for the developers to build programs in a specific platform. These frameworks include predefined classes and functions that can be used to process the input and to manage the hardware devices, and to interact with system software. It is a toolset used as innovative, cost- effective solutions to the problems in big

data processing and helps to provide an insight knowledge incorporating metadata for taking decision to the business needs.

There are many frameworks available for big data as a open source that can be used instantly in order to process the Big data. Some of the popular and widely used open source Big Data frameworks [2] are Hadoop, Spark, Storm, Flink etc. These frameworks are developed by Apache Company in order to process the big data. Some score high utility and some have high potential. Apache Flink is one of the popular open source frameworks that are used for both batch and stream data processing [12]. It is expressive, declarative, fast and efficient data analysis for both batch and real time data. Apache Flink reduces the Complexity faced by the distributed data driven engines. Fig 1 shows the usage of Flink in both It combines the scalability and programming flexibility of MapReduce. It is suited for cluster environments. It is 100 time faster than Hadoop- MapReduce. Flink offers a reduced level of complexity by integrating the traditional database concepts. It also offers a very high level of scalability. In this paper the architecture is discussed briefly in section 2. The library functions and interfaces are mentioned in section 3. The main features that make the Flink Popular are listed in detail in section 4. Various other frameworks are compared with Flink in section 5. Finally the conclusion is made in section 7.

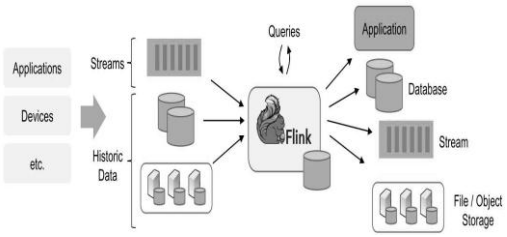

Fig 1: Basic Usage of Apache Flink

2. ARCHITECTURE

Flink is a distributed system that requires allocation and management of resources that will execute streaming application. Common Clusters like Hadoop YARN, Apache Mesons resource manager integrates in this and also it can be set as a standalone cluster [6].

The Flink consist of two processes [6]: Job Manager and Task Managers. Fig [2] shows the architecture of Flink. The Job Manager is used to coordinate the Flink System, and the Task managers are the workers that will executer the parts as parallel program. When the system is stared in local mode a Single Job and Task Manager are used within the same JVM. When a program is submitted it performs the preprocessing and turns it into a parallel data that execute by Job Manager and Task Manager[8].

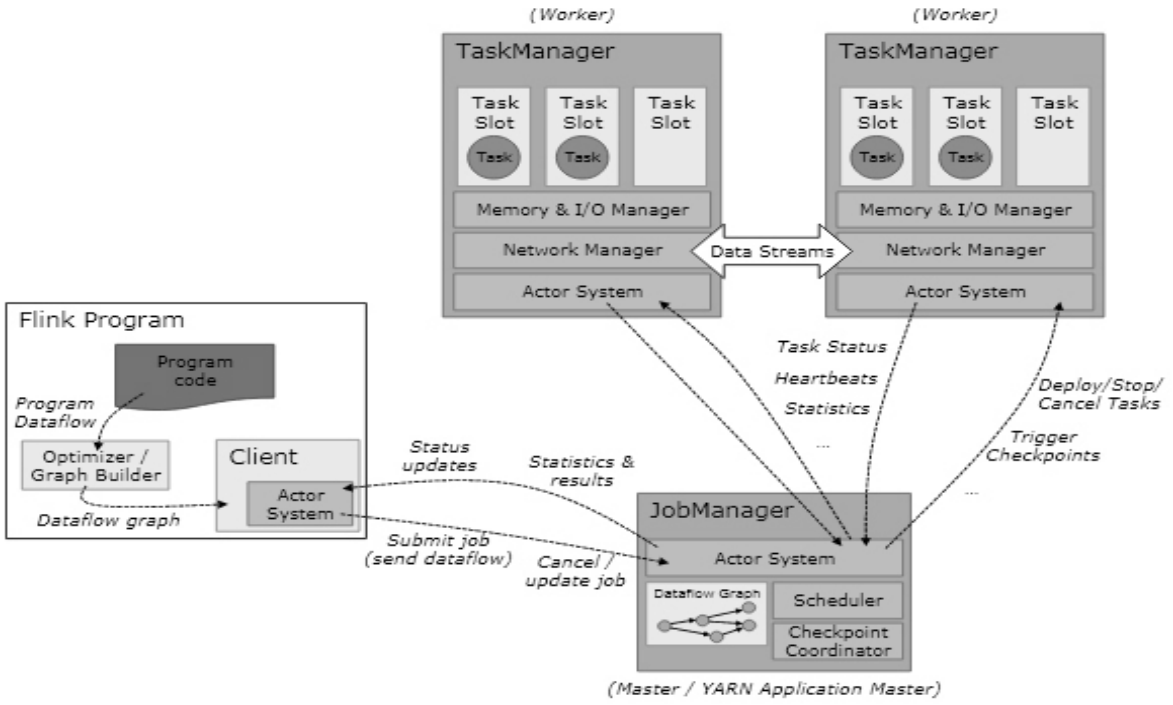

Fig 2: Basic Usage of Apache Flink

Job Manager has number of responsibilities based on coordinating the distributed execution. It has to schedule the next task, analysis the finish task or any failure, monitor checkpoints and coordinates recovery. There are three components, they are *Resource manager*: it is responsible for resource allocation and de-allocation in a cluster, it also mange task slots. It implements multiple Resource Manager for different environment this only distribute the slots for Task Manager. *Dispatcher* provides a interface to submit application for execution and starts new Job Master. *Job Master:* it is responsible for managing the execution of a single Job Graph. Task Managers execute the task of a dataflow and buffer and exchange the data streams. The smallest unit of resource scheduling is called as slot. The task slot present indicates concurrent processing task.

Each worker may execute one or more subtask in separate thread. To handle the task a Task Manger uses task slots [6]. This represents a fixed subset of resources. By adjusting the number of slots, it can define how subtasks are isolate from each other. Flink allows subtasks to share slots even they are in different task. So one slot can hold an entire pipeline of the job. There are two benefits for allowing the slots to share.

2.1 STREAM PROCESSING

Flink supports Stream processing [6] of data it can be from web server, stock exchange or sensor reading from machine. For analyzing these data by organize the processing by bounded or unbounded streams.

Batch processing, works when a bounded data stream is processed. It will sort the data, computer statistics and produce final report that summarizes all the input. Stream processing involves unbounded data stream. The arrival of input never ends do it has to continuously process the data when it arrives. Flink uses streaming dataflow that transformed by user-defined operator. These dataflow form directed graph from sources and end in sinks

as shown in Fig [3]. The transformation is a one to one correspondence that may consist of multiple operators.

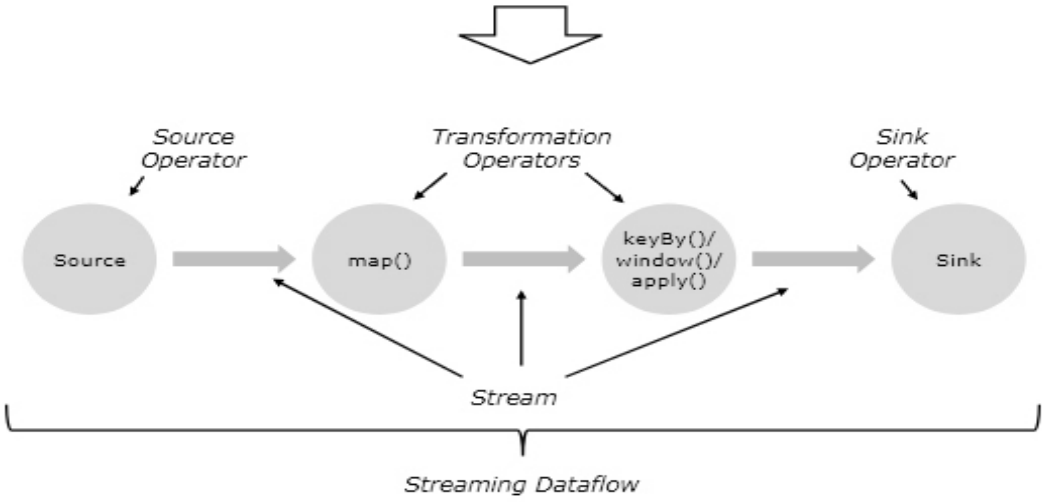

Fig 3: A skeleton for Streaming Dataflow

3. DATASTREAM API

The program for Data stream is a normal program the transform the data. The data is created initially from various sources as shown in Fig [1]. The results are returned through sinks it is the standard output. The DataStream API is the name form class DataStream used to represent data collection. This data can be finite or unbounded. It is similar to Java Collection [6], immutable means once create the elements cannot be add or remove but it can be transformed by using DataSream API operations. To write a Flink DataStream program, an anatomy has to be followed and stream transformation operators can be added to transform the source data. The basic steps followed for writing a program are [6]

1. Obtain an Execution Environment, 2.Load/create the Initial Data 3. Specify transformations on the data 4.Specify where the result has to be placed 5.Trigger the execution.

3.1. LIBRARIES FOR DATA STREAMING API

Data Sources

The program read the input from any device or application by using *StreamExecutionEnvironment.addSource(SourceFunction).* There are several Stream functions present they are: *File-based, Socket-based, Collection-based* and *Custom*.

DataStream Transformation

Operators transform the data into a new DataStream. Some of them are Map, *FlatMap, Filter, KeyBy, Iterate, Connect, Union, Reduce, Window Reduce* etc. The transformation can be done based on physical partitioning some of them are Custom partitioning, Random partition, Rescaling, Rebalancing and Broadcasting.

Data Sinks

This is the final stage the DataStream's are got and forward to files, socket, external system or print it. It uses variety of output formats such as *writeAsText(), writeCsv(..), print(), writeUsingOUtputFormat(), writeToSocket* and *addsink.*

4. KEY FEATURES

Some of the main features of Apache Flink are

a) True Streaming Engine:

Streaming data enables [9] a new type of latency-critical application and giver more operational insight. It is simple and flexible. A stream processor can deliver high throughput(pushing large amount of data through the pipeline) , low latency and strong consistency, Low overhead(fault tolerance mechanism) in the presence of Stateful computations since the computation is powerful. It is a scalable stream processing engine.

b) Custom Memory manager

It has own memory management inside the JVM. The main characteristics are 1. User data stored in Serialized bytes array. 2. C++ style memory management.3. The allocation and de-allocation for implementing memory is done on internal buffer pool. The advantage of this is no Exception arises, Reduce the garbage collection, no need for runtime tuning, it is more reliable and performance is stable.

c) Native closed-loop iteration operators

It uses iterative computations. The algorithm is tightly bounded into Flink query optimizer. The pipelined structure processes the data faster with low latency.

d) Automatic cost-based optimizer

The batch programs are optimized automatically so the expensive operations like shuffle and sort can be avoided.

e) Easy to use

API makes the streaming data to use easily than the Programming of MapReduce and the testing can be done easier when compared to Hadoop.

5. COMPARISON WITH VARIOUS FRAMEWORKS

Apache Flink has various advantages over other frameworks some of the key features are compared with the other frameworks [9] like Spark, and Storm [1] .

The Comparison of Apache Flink with other framework like Apache Spark [11] and Apache Strom [7] are compared and listed in Table 1.

Table 1: Comparison Study of Flink with Spark and Storm

Features	**Apache Flink**	**Apache Spark**	**Apache Storm**
Computation Model	Operator Based Model	Micro-batched Model	micro-batching using a trident
Streaming Engine	Streams are used for all workload	Micro-batch are used for all workload	Graph Streaming is used.
Iterative Processing	Iteration operations are use, (Iterate and Delta Iterate)	Non Native Iteration, implemented as regular for-loops	Iteration is not predefined
Optimization	The optimizer is independent with the programming interface.	It is manually optimized.	Optimized manually
Latency	Low latency and high throughput	High latency	better latency with fewer restrictions
Performance	Overall performance	Performance is	Improve the

	is excellent	excellent but only for micro-batch processing	performance
Fault Tolerance	Mechanism is lightweight so maintain high throughput rates and strong consistency	Recovers last work and deliver only once.	Restarts automatically.
Speed	Lightening fast speed	Slower than Flink	Slower than Flink

6. CONCLUSION

Flink Simplifies the parallel analysis of large amounts of data. This can be used for both batch and real time data. A brief review is taken in various criteria about Apache Flink and reviewed properly to get a clear knowledge about the how to use the Flink Framework. The Flink can handle distributed dataflow in a reliable way and it improves the topology frame. So the performance is increased. Flink employs streaming data and batch processing in a same streaming engine. The resource utilization and execution scalability is good for the number of cluster volumes. Since the Apache Flink framework is very popular framework for processing streaming data.

7. REFERENCE

[1] Mrs. Tanuja Pattanshetti, Mr. Subodh Kamble, Mr. Aditya Yalgude, Mr. Pranav Patil," A survey on 'Apache Storm performance optimization using tuning of parameters" 11th ICCCNT 2020

[2] Safaa Alkatheri1, Samah Anwar Abbas, Muazzam Ahmed Siddiqui, "A Comparative Study of Big Data Frameworks" International Journal of Computer Science and Information Security (IJCSIS), Vol. 17, No. 1, ISSN 1947-5500, January 2019

[3] N. Deshai, B.V.D.S. Sekhar, S. Venkataramana, "Processing Big Data with Apache Flink", International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8 Issue-1S3, June 2019.

[4] Ankush Arunrao Parise, Rupali Ganesh Rajurkar "Big Data: Tools & Applications", International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056, Volume: 06 Issue: 02,Feb 2019.

[5] Alcalde-Barros, A., García-Gil, D., García, S. "DPASF: a flink library for streaming data preprocessing" Springer nature. *Big Data Anal* **4,** 4 (2019)

[6] Sachini Jayasekara, Shanika Karunasekera and Aaron Harwood "Enhancing the Scalability and Performance of Iterative Graph Algorithms on Apache Storm" , IEEE International Conference on Big Data (Big Data),2018.

[7] Tilmann Rabl, Jonas Traub, and Volker Markl, "Apache Flink in Current Research Projects", it – Information Technology 2016; 58(4): 157–165, May 7 2016.

[8] Sanket Chintapalli, Derek Dagit, Bobby Evans, Reza Farivar, Thomas Graves, Mark Holderbaugh Zhuo Liu, Kyle Nusbaum, Kishorkumar Patil, Boyang Jerry Peng and Paul Poulosky "Benchmarking Streaming Computation Engines: Storm, Flink and

Spark Streaming",IEEE International Parallel and Distributed Processing Symposium Workshops,2016.

[9] Asterios Katsifodimos , Sebastian Schelter "Apache Flink: Stream Analytics at Scale ",IEEE International Conference on Cloud Engineering Workshop,2016.

[10] Ramkrushna C. Maheshwar, D. Haritha," Survey on High Performance Analytics of Bigdata with Apache Spark" International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), ISBN No.978-1-4673-9545-8,2016.

[11] Paris Carbone Asterios Katsifodimos, Stephan Ewen Volker Markl, Seif Haridi Kostas Tzoumas "Apache Flink™: Stream and Batch Processing in a Single Engine" IEEE Computer Society Technical Committee on Data Engineering", 2015.