FUNDATEC
FUNDACION TECNOLOGICA DE COSTA RICA

Especialidad Data Science

Curso:

Big Data

Profesor:

Felipe Meza

Estudiante:

Sergio Castillo Segura

Tarea #4

Micro investigación sobre Data Streaming Utilizando Apache Spark

Lunes 20 Enero, 2020

FlumeJava

MapReduce es la técnica de procesamiento digital en paralelo que permite analizar grandes cantidades de datos e información, aprovecha recursos compartidos de todo tipo para hacer distribución de las cargas de trabajo y es muy eficiente a la hora de administrar errores y consolidar los resultados de los distintos “trabajadores”.

Sin embargo, la técnica de MapReduce por defecto solo administra la ejecución de un trabajo de procesamiento a la vez, y cuando es necesario realizar todo un flujo de ejecución en el cuál el resultado de una tarea MapReduce debe ser transferido a una nueva tarea y así sucesivamente se vuelve mucho más complejo y requiere de mucho trabajo de enlace manual vía programática.

Estos flujos o “pipelines” son muy comunes en el mundo de big data y por tanto era necesaria una evolución.

En Google se dieron a la tarea de crear FlumeJava, que es una librería para Java basada en MapReduce que permite administrar de forma fácil y escalable los pipelines de MapReduce.

Funcionamiento

FlumeJava se encarga de crear planes de ejecución para las tareas encadenadas, optimizando el flujo de los datos de modo que el procesamiento sea continuo y en paralelo.

Utiliza Evaluaciones Diferidas para arrancar la ejecución más rápido, tiene un Optimizador para los planes definidos por el usuario, y tiene un Ejecutor que se encarga de ejecutar cada batch de procesamiento independiente.

Una de las desventajas actuales es que aún no cuenta con soporte para pipelines incrementales o de ingesta de datos en Stream. Además el optimizador no evalúa el código de usuario.

Flink Streaming

Apache Flink es un motor de procesamiento distribuido para sets de datos finitos e infinitos manteniendo el estado del procesamiento en todo momento (stateful). Puede ejecutarse sobre todos los ambientes de clúster comunes como YARN, Mesos y Kubernetes, y es ultra rápido, eficiente y simple de utilizar.

Su base de trabajo podría asemejarse a la de Apache Spark, la diferencia es que Spark está optimizado para trabajar sobre micro-batches de datos finitos y cualquier procesamiento de datos en tiempo real se debe hacer utilizando complementos externos manualmente optimizados, Flink sin embargo, su naturaleza es el procesamiento en tiempo real donde todos sus elementos están optimizados por defecto para este modo de trabajo, resultando en mayor agilidad y menor latencia.

Funcionamiento

Flink puede ser implementado en sistemas distribuidos para procesamiento en paralelo y aprovechando todos los recursos disponibles, pero también puede ejecutarse en una sola máquina. El estado de todas las tareas en ejecución siempre se mantiene en memoria o en almacenamiento permanente de lectura optimizada, optimizando latencias. En caso de utilizar memoria volátil, Flink asegura el acceso al estado actual de cualquier stream de datos al hacer descargar frecuentes y asíncronas a memoria permanente.

El acceso a Flink siempre es a través de APIs, teniendo 3 capas principales:

- Alto nivel para analíticos: provee una interface para ejecución de sentencias SQL.
- Procesamiento de Datos en Stream y Batch: por medio de punteros Streams y ventanas de memoria.
- Aplicaciones impulsadas por eventos stateful: al estilo callbacks y promesas para events, estados y tiempo.

Beam Streaming

Beam es una capa de abstracción para tareas de procesamiento paralelo tanto en modo Batch como en modo Streaming.

Beam surgió de la necesidad de facilita el trabajo de migración de las tareas convencionales escritas para MapReduce hacia otros frameworks más eficientes como Spark o Flink. Si estas tareas se hacen de modo manual normalmente resultan en la rescritura completa del código, en cambio Beam funciona como un SDK para la escritura del programa y para su ejecución, de modo que migrar la tarea de un ambiente de ejecución a otro sea una labor mucho más sencilla.

Actualmente Beam soporta muchos tipos de ambientes de ejecución (backends), entre los cuáles se listan: Apex, Flink, Gearpump, Saamza, Spark, Google Dataflow y Hazelcast Jet.

Y el SDK está disponible tanto para Scala, Java, Python y Go.

Google fue uno de sus desarrolladores junto con Cloudera y Paypal, quienes donaron el software a la comunidad open source por medio de Apache.

Funcionamiento

Es sencillo, utilizando el SDK unificado de Beam para el lenguaje de preferencia se escribe el programa que defina el pipeline de procesamiento, Beam usa las mismas clases e interfaces tanto para Batch como Stream facilitando el proceso, listo esto el programa es traducido y optimizado por Beam hacia el API del ambiente de ejecución deseado.

Google Dataflow

El Big Data no es un tema sencillo, en realidad puede resultar bastante complejo, para poder procesar una gran cantidad de datos de forma eficiente y en el tiempo adecuado se requieren recursos y administración de ambientes de ejecución que requieren mucho mantenimiento y optimización. Es por esto que existen también servicios en la nube que son completamente automatizados para aprovisionar los recursos necesarios y ejecutar los pipelines de procesamiento según las necesidades específicas, Dataflow es uno de ellos provisto por medio de Google Cloud.

Funcionamiento

Dataflow es un servicio elástico y distribuido en la nube de ejecución de datos masiva en paralelo tanto para modo Batch como Streaming. Provee toda la automatización necesaria para aprovisionar los recursos indicados al inicio de la ejecución como para el escalamiento horizontal para la agilización del procesamiento de los datos.

Provee todos los servicios de MapReduce así como de micro-batches y Streaming, es compatible con Apache Flink de modo tal que se pueden escribir tareas genéricas utilizando el SDK para ejecutar de forma local como en la nube.

A su vez, provee interfaces para analíticos tipo SQL, servicios de monitoreo y notificación de problemas, plantillas para compartir las tareas entre equipos de trabajo, llaves de encriptación de datos, IPs privadas, entre otros.

De esta forma Dataflow es una solución para procesamiento de datos cuando no se disponen los recursos necesarios para montar un cluster de procesamiento, o cuando se requiere un resultado inmediato sin tener que perder tiempo configurando un clúster de ejecución.