APACHE SPARK

Spark es un motor informático distribuido unificado en diferentes cargas de trabajo y plataformas. Spark puede conectarse a diferentes plataformas y procesar diferentes cargas de trabajo de datos mediante una variedad de paradigmas como Spark streaming, Spark ML, Spark SQL y Spark GraphX.

Apache Spark es un motor de procesamiento de datos rápido en memoria con API de desarrollo elegantes y expresivas para permitir a los trabajadores de datos ejecutar de manera eficiente el aprendizaje automático de streaming o cargas de trabajo SQL que requieren un acceso interactivo rápido a conjuntos de datos. Apache Spark consta de núcleo de Spark y un conjunto de bibliotecas.

El núcleo es el motor de ejecución distribuido y las API de Java, Scala y Python ofrecen una plataforma para el desarrollo de aplicaciones distribuidas. Las bibliotecas adicionales creadas sobre el núcleo permiten cargas de trabajo para streaming, SQL, procesamiento de gráficos y aprendizaje automático. Spark ML, por ejemplo, está diseñado para la ciencia de datos y su abstracción facilita la ciencia de datos.

Spark proporciona streaming en tiempo real, consultas, aprendizaje automático y procesamiento de gráficos. Antes de Apache Spark, teníamos que utilizar diferentes tecnologías para diferentes tipos de cargas de trabajo, una para análisis por lotes, otra para consultas interactivas, otra para el procesamiento de streaming en tiempo real y otra para algoritmos de aprendizaje automático. Sin embargo, Apache Spark puede hacer todo esto simplemente usando Apache Spark en lugar de usar múltiples tecnologías que no siempre están integradas.

Nota

Con Apache Spark, se pueden procesar todos los tipos de carga de trabajo y Spark también es compatible con Scala, Java, R y Python como medio de escribir programas cliente.

Spark es un motor informático distribuido de código abierto que tiene ventajas clave sobre el paradigma MapReduce:

1. Utiliza el procesamiento en memoria tanto como sea posible
2. Motor de uso general que se utilizará para cargas de trabajo por lotes y en tiempo real
3. Compatible con YARN y también Mesos
4. Se integra bien con HBase, Cassandra, MongoDB, HDFS, Amazon S3 y otros sistemas de archivos y fuentes de datos

Spark es una plataforma para distribuida que cuenta con varias características:

1. Procesa de forma transparente los datos de varios nodos a través de una API sencilla
2. Maneja de forma resistente los errores
3. Derrame los datos en el disco según sea necesario, aunque utiliza predominantemente memoria
4. Se admiten las API de Java, Scala, Python, R y SQL
5. El mismo código Spark puede ejecutarse de forma independiente, en Hadoop YARN, Mesos y la nube.

Nota

Las características de Scala como implícitas, funciones de orden superior, tipos estructurados, etc., nos permiten crear fácilmente DSL e integrarlos con el lenguaje.

Apache Spark no proporciona una capa de almacenamiento y se basa en HDFS o Amazon S3, etc. Por lo tanto, incluso si las tecnologías Apache Hadoop se reemplazan por Apache Spark, HDFS todavía es necesario para proporcionar una capa de almacenamiento confiable.

Nota

Apache Kudu ofrece una alternativa a HDFS y ya hay integración entre Apache Spark y la capa Kudu Storage, desacoplando aún más Apache Spark y el ecosistema de Hadoop.

Hadoop y Apache Spark son marcos de big data populares, pero realmente no sirven para los mismos propósitos. Aunque Hadoop proporciona almacenamiento distribuido y un marco informático distribuido MapReduce, Spark, por otro lado, es un marco de procesamiento de datos que funciona en el almacenamiento de datos distribuido proporcionado por otras tecnologías.

Spark es generalmente mucho más rápido que MapReduce debido a la forma en que procesa los datos. MapReduce opera en divisiones mediante operaciones de disco, Spark opera en el conjunto de datos de forma mucho más eficiente que MapReduce, con la razón principal detrás de la mejora del rendimiento en Apache Spark es el eficiente procesamiento fuera del montón en memoria en lugar de depender únicamente de cálculos basados en disco.

Nota

El estilo de procesamiento de MapReduce puede ser suficiente si las operaciones de datos y los requisitos de informes son en su mayoría estáticos y está bien usar el procesamiento por lotes para sus propósitos, pero si necesita realizar análisis en datos de streaming o sus requisitos de procesamiento necesitan lógica de procesamiento multietapa, probablemente desee ir con Spark.

Hay tres capas en la pila de Spark. La capa inferior es el administrador de clústeres, que puede ser independiente, YARN o Mesos.

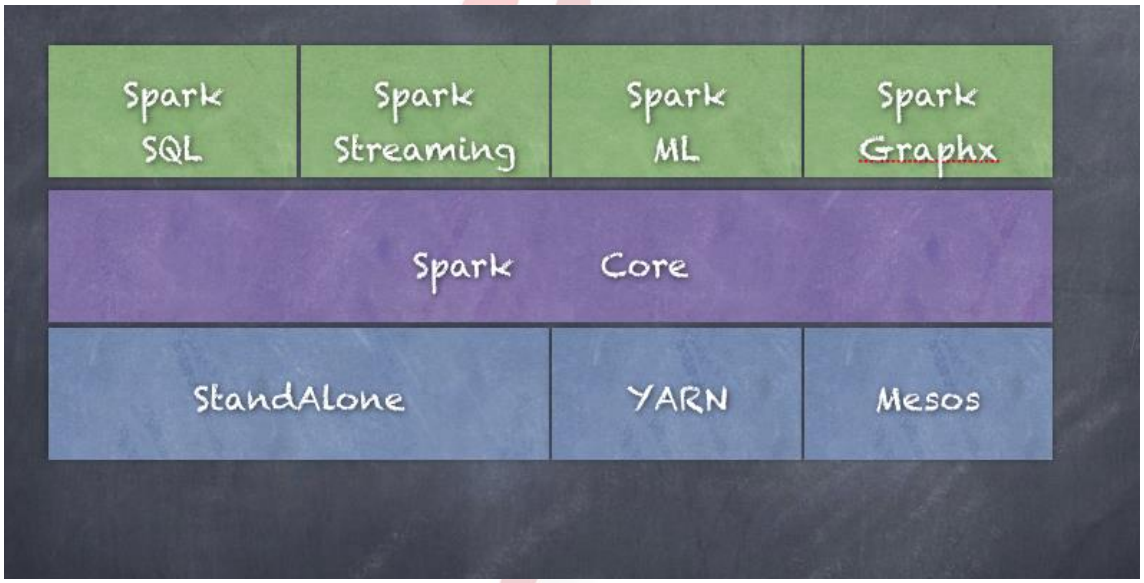
Nota

Con el modo local, no necesita un administrador de clústeres para procesar.

En el centro, encima del administrador de clústeres, se encuentra la capa de núcleo de Spark, que proporciona todas las API subyacentes para realizar la programación de tareas e interactuar con el almacenamiento.

En la parte superior hay módulos que se ejecutan sobre el núcleo de Spark, como Spark SQL, para proporcionar consultas interactivas, streaming de Spark para análisis en tiempo real, Spark ML para aprendizaje automático y Spark GraphX para el procesamiento de gráficos.

Las tres capas son las siguientes:



Como se ve en el diagrama anterior, las distintas bibliotecas como Spark SQL, Spark streaming, Spark ML y GraphX se encuentran encima del núcleo de Spark, que es la capa intermedia. La capa inferior muestra las distintas opciones del gestor de clústeres.

Veamos ahora cada uno de los componentes brevemente:

Spark Core

Core es el motor de ejecución general subyacente para la plataforma Spark en el que se basa toda la otra funcionalidad. Spark Core contiene funcionalidades básicas de Spark necesarias para ejecutar trabajos y necesarias para otros

componentes. Proporciona conjuntos de datos de referencia y computación en memoria en sistemas de almacenamiento externos, siendo el importante el conjunto de datos distribuido resistente (RDD).

Además, Spark Core contiene lógica para acceder a varios sistemas de archivos, como HDFS, Amazon S3, HBase, Cassandra, bases de datos relacionales, etc. Spark Core también proporciona funciones fundamentales para admitir redes, seguridad, programación y barajado de datos para crear una plataforma altamente escalable y tolerante a errores para la informática distribuida.

Los marcos de datos y conjuntos de datos creados sobre RDD e introducidos con Spark SQL se están convirtiendo en la norma ahora sobre rdDs en muchos casos de uso. Los RDD son aún más flexibles en términos de manejo de datos totalmente no estructurados, pero en conjuntos de datos futuros, la API podría convertirse en la API principal.

Spark SQL

Spark SQL es un componente sobre el núcleo de Spark que una nueva abstracción de datos denominada SchemaRDD, que proporciona compatibilidad con datos estructurados y semiestructurados. Spark SQL proporciona la manipulación de grandes conjuntos de datos distribuidos y estructurados mediante un subconjunto SQL admitido por Spark y Hive QL. Spark SQL simplifica el control de datos estructurados a través de DataFrames y conjuntos de datos en un nivel mucho más eficaz como parte de la initativa de tungsteno.

Spark SQL también admite la lectura y escritura de datos en y desde varios formatos estructurados y orígenes de datos, archivos, parquet, orc, bases de datos relacionales, Hive, HDFS, S3, etc. Spark SQL proporciona un marco de optimización llamado Catalyst para optimizar todas las operaciones para aumentar la velocidad (en comparación con RDDs Spark SQL es varias veces más rápido). Spark SQL también incluye un servidor Thrift, que pueden ser utilizados por sistemas externos para consultar datos a través de Spark SQL mediante protocolos JDBC y ODBC clásicos.

Spark Streaming

Spark Streaming por secuencias aprovecha la capacidad de programación rápida del núcleo de Spark para realizar análisis de streaming mediante la ingesta de datos de streaming en tiempo real de varias fuentes, como HDFS,

Kafka, Flume, Twitter, ZeroMQ, Kinesis, etc. El streaming de Spark utiliza microtejos de datos para procesar los datos en fragmentos y, usa un concepto conocido como DStreams, la transmisión por secuencias de Spark puede funcionar en los RDD, aplicando transformaciones y acciones como RDD normales en la API principal de Spark.

Las operaciones de streaming de Spark pueden recuperarse de errores automáticamente mediante varias técnicas. La transmisión por secuencias de Spark se puede combinar con otros componentes de Spark en un único programa, unificando el procesamiento en tiempo real con el aprendizaje automático, SQL y las operaciones de gráficos.

Además, la nueva API de streaming estructurado hace que los programas de streaming de Spark sean más similares a los programas por lotes de Spark y también permite realizar consultas en tiempo real sobre los datos de streaming, lo que es complicado con la biblioteca de streaming de Spark antes de Spark 2.0+.

Spark GraphX

Es un marco de procesamiento de gráficos distribuidos en la parte superior de Spark. Los gráficos son estructuras de datos que comprenden vértices y los bordes que los conectan. GraphX proporciona funciones para crear gráficos, representados como RDD de gráfico. Proporciona una API para expresar el cálculo de gráficos que puede modelar gráficos definidos por el usuario mediante la API de abstracción de Pregel. También proporciona un tiempo de ejecución optimizado para esta abstracción. GraphX también contiene implementaciones de los algoritmos más importantes de la teoría de gráficos, como el rango de páginas, los componentes conectados, las rutas más cortas, SVD++ y otros.

Un módulo más reciente conocido como GraphFrames está en desarrollo, lo que facilita el procesamiento de gráficos mediante gráficos basados en DataFrame. GraphX es para RDDs lo que GraphFrames son para DataFrames/datasets. Además, actualmente es independiente de GraphX y se espera que admita toda la funcionalidad de GraphX en el futuro, cuando podría haber un cambio a GraphFrames.

Spark ML

es un marco de aprendizaje automático distribuido por encima de Spark core maneja modelos de aprendizaje automático utilizados para transformar conjuntos de datos en forma de RDD. Spark MLlib es una biblioteca de algoritmos de aprendizaje automático que proporciona varios algoritmos como la regresión logística, clasificación de Bayes naive, máquinas vectoriales de soporte (SVM), árboles de decisión, bosques aleatorios, regresión lineal, cuadrados mínimos alternativos(ALS)y clustering k-means. Spark ML se integra muy bien con Spark core, Spark streaming, Spark SQL y GraphX para proporcionar una plataforma verdadera donde los datos pueden ser en tiempo real o por lotes.

Además, **PySpark** y **SparkR** también son como medios para interactuar con clústeres de Spark y utilizar las API de Python y R. Las integraciones de Python y R realmente abren Spark a una población de científicos de datos y modeladores de aprendizaje automático, ya que los lenguajes más comunes utilizados por los científicos de datos en general son Python y R. Esta es la razón por la que Spark admite la integración de Python y también la integración de R, con el fin de evitar el costoso proceso de aprendizaje de un nuevo lenguaje de Scala. Otra razón es que puede haber una gran cantidad de código existente escrito en Python y R, y si podemos aprovechar parte del código, eso mejorará la productividad de los equipos en lugar de construir todo de nuevo desde cero.

Hay una creciente popularidad y uso de tecnologías de portátiles como Jupyter y Zeppelin, que hacen que sea significativamente más fácil interactuar con Spark en general, pero particularmente muy útil en Spark ML, donde se esperan muchas hipótesis y análisis.

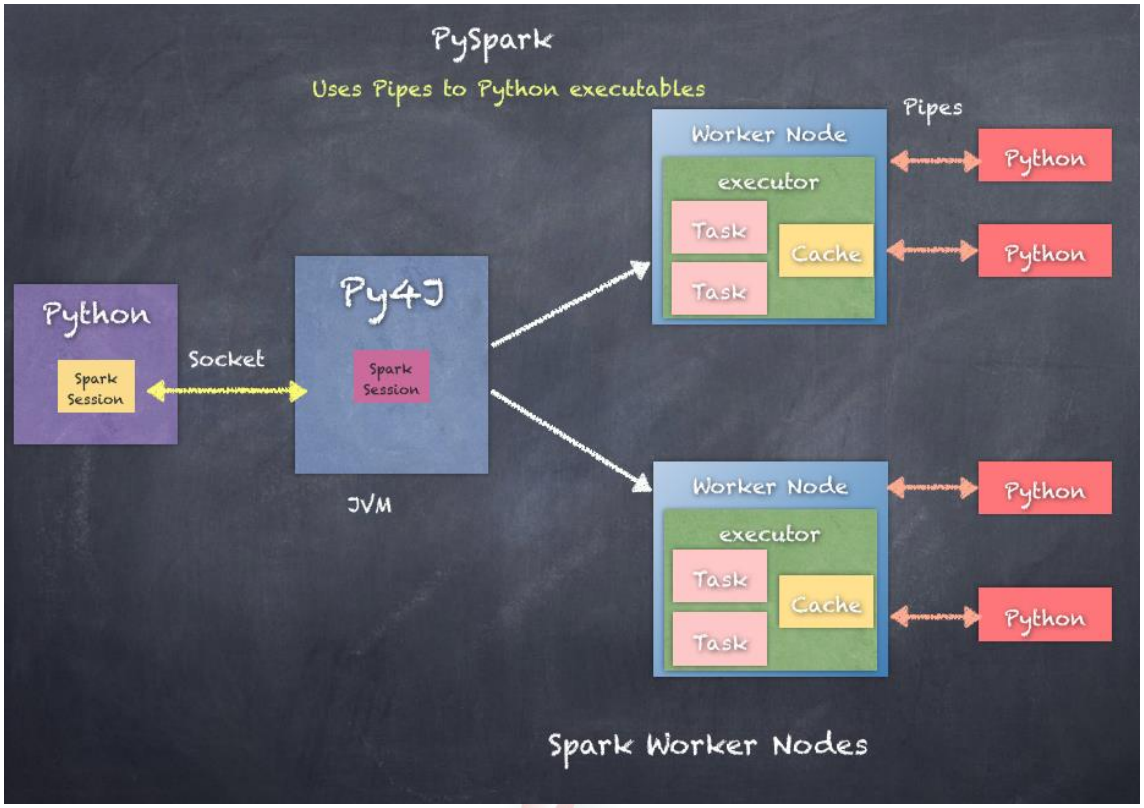
PySpark

utiliza scripts basados en Python y Python como tareas y, a continuación, utiliza sockets y canalizaciones para ejecutar procesos para comunicarse entre clústeres de Spark basados en Java y scripts de Python. PySpark también utiliza, que es una biblioteca popular integrada dentro de PySpark que permite la interfaz de Python dinámicamente con RDD basados en Java.SparkContextPy4J

Nota

Python debe estar instalado en todos los nodos de trabajo que ejecutan los ejecutores de Spark.

A continuación, **se muestra cómo funciona PySpark comunicándose entre scripts Java procesados y Python:**



SparkR

SparkR es un paquete de R que un front-end ligero para usar Apache Spark de R. SparkR proporciona una implementación de marco de datos distribuido que admite operaciones como selección, filtrado, agregación, etc. SparkR también admite el aprendizaje automático distribuido mediante MLlib.

SparkR utiliza scripts basados en R y R como tareas y, a continuación, utiliza JNI y canalizaciones para ejecutar procesos para comunicarse entre clústeres de Spark basados en Java y scripts de R.SparkContext

Nota

R debe instalarse en todos los nodos de trabajo que ejecutan los ejecutores de Spark.

A continuación, se muestra cómo funciona SparkR comunicándose entre scripts Java procesados y de R: